ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

MÉMOIRE PRÉSENTÉ À
L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

COMME EXIGENCE PARTIELLE
À L'OBTENTION DE LA
MAÎTRISE EN GÉNIE DES TECHNOLOGIES DE L'INFORMATION
M.Ing.

PAR
Hicham LAYACHI

SYSTÈME UNIFIÉ DE TRANSCODAGE VIDÉO PERMETTANT LA RÉALISATION D'UNE COMBINAISON D'OPÉRATIONS DE TRANSCODAGE

MONTRÉAL, LE 08 SEPTEMBRE 2010

CE MÉMOIRE A ÉTÉ ÉVALUÉ
PAR UN JURY COMPOSÉ DE :

M. Stéphane Coulombe, directeur de mémoire
Département de génie logiciel et des TI à l'École de technologie supérieure

Mme Rita Noumeir, présidente du jury
Département de génie électrique à l'École de technologie supérieure

M. Pierre Dumouchel, membre de jury
Département de génie logiciel et des TI à l'École de technologie supérieure

IL A FAIT L'OBJET D'UNE SOUTENANCE DEVANT JURY ET PUBLIC

LE 24 AOUT 2010

À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

SYSTÈME UNIFIÉ DE TRANSCODAGE VIDÉO PERMETTANT LA RÉALISATION D'UNE COMBINAISON D'OPÉRATIONS DE TRANSCODAGE

Hicham LAYACHI

RÉSUMÉ

Au fil des années, l'utilisation de la vidéo dans la vie quotidienne ne cesse d'augmenter, surtout grâce à l'explosion d'Internet et l'usage massif des différentes sortes d'appareils mobiles : téléphones cellulaires, assistants personnels numériques (*PDA*), téléphones intelligents, etc. Ces derniers ont nettement surpassé, en nombre, les ordinateurs personnels. L'hétérogénéité des réseaux (filaires, sans-fils, fibres optiques,...), la diversité des appareils ainsi que la variété des applications multimédias utilisées rendent indispensable l'adaptation du contenu multimédia afin de permettre un accès universel et transparent aux usagers finaux.

Le transcodage vidéo est une technologie inévitable qui permet de répondre à ce problème en transformant la vidéo encodée selon les nouveaux besoins de transmission et/ou de l'usager. Le transcodage est devenu donc un sujet de recherche très actif où plusieurs architectures et systèmes de transcodage ont été proposés pour les différents cas d'utilisation : adaptation du débit binaire, adaptation de la résolution spatiale, adaptation de la résolution temporelle, adaptation du format (standard), insertion de logo, etc.

Cependant, la majorité des solutions proposées sont le résultat d'études de ces cas d'utilisation de transcodage vidéo pris séparément. Dans ce projet, nous analysons les différentes architectures vidéo présentées dans la littérature et nous proposerons une architecture unifiée permettant de réaliser efficacement différentes combinaisons d'opérations de transcodage dans un seul système. Ensuite nous proposons des variantes aux algorithmes de transcodage existants qui sont mieux adaptées à notre système. Ce nouveau système de transcodage est implémenté et les performances validées à l'aide de simulations sur plusieurs combinaisons d'opérations de transcodage. Nos résultats montrent que des gains permettant de supporter 2.6 fois plus de canaux, en moyenne sur un processeur Intel, sont obtenus pour l'adaptation du débit binaire de 512 à 384 kb/s avec une perte mineure en PSNR de -0.12 dB et des gains de 2.14 en moyenne pour l'adaptation de la résolution spatiale de 512 à 256 kb/s avec une perte de -0.95 dB. Pour l'adaptation de format de 512 à 512 kb/s, nos résultats montrent des gains de 1.84 fois plus de canaux en moyenne avec une perte en PSNR de -1.14 dB et pour l'adaptation de format avec adaptation de la résolution spatiale de 256 à 32 kb/s des gains de 1.19 en moyenne avec une perte -0.41 dB ont été obtenus.

Mots-clés : transcodage, hétérogène, adaptation du débit binaire, adaptation de la résolution spatiale, adaptation de la résolution temporelle, adaptation du format, insertion de logo.

UNIFIED VIDEO TRANSCODING SYSTEM TO REALIZE MULTIPLE VIDEO TRANSCODING OPERATIONS

Hicham LAYACHI

ABSTRACT

Over the years, the use of the video in our daily life has been increasing, thanks to the internet and the various sorts of mobile devices: cellular telephones, personal digital assistants (PDA), smartphones, etc. These devices have already exceeded the number of personal computers. The heterogeneity of networks (cable, wireless networks, optical fibers, etc.), the variety of devices as well as the various deployed multimedia applications have made indispensable the adaptation of the multimedia contents in order to facilitate universal and transparent access to the final users.

Video transcoding is a key technology which allows mitigating this problem by transforming the previously encoded video according to the new transmission and/or user needs. Thus, transcoding has become a very active research area in which several transcoding architectures and systems have been proposed for various use cases such as bitrate adaptation, spatial resolution adaptation, temporal resolution adaptation, format adaptation, logo insertion, etc.

However, the majorities of proposed solutions were standalone and have concentrated on the study of various video transcoding operations separately. In this project, we analyze the various video architectures presented in the literature and propose a unified architecture allowing the realization of various transcoding operations addressing a large number of use cases in a single system. Then, we propose variants to the existing video transcoding algorithms which are better adapted to our system. In order to evaluate the proposed unified video transcoding system, experiments are performed for different combinations of video transcoding operations. Our results show that the proposed system permits an average of 2.6 more channels with negligible quality loss of 0.12 dB for the bitrate adaptation from 512 to 384 kb/s. For the spatial resolution adaptation from 512 to 256 kb/s, an average of 2.14 more channels with 0.95 dB loss is obtained. For the format adaptation from 512 to 512 kb/s, the obtained results show an average of 1.84 times more channels with quality loss of 1.14 dB. Finally, for the format adaptation with spatial resolution adaptation from 256 to 32 kb/s, an average of 1.19 more channels with 0.41 dB loss is obtained.

Keywords: transcoding, heterogeneous, bitrate adaptation, spatial resolution adaptation, temporal resolution adaptation, format adaptation, logo insertion.

REMERCIEMENTS

Je tiens à remercier, vivement, mon directeur de recherche, M. Stéphane Coulombe, pour son soutien, ses conseils et ses remarques judicieux durant toutes les phases de ma maîtrise.

Je tiens aussi à remercier M. Steven Pigeon pour sa précieuse aide et ses conseils pertinents.

Je remercie également notre partenaire Vantrix Inc. qui m'a offert l'opportunité de réaliser ce projet de recherche ainsi que tous mes collègues de l'équipe MARS (Multimedia Adaptation Research and Systems) pour leur support et encouragement.

Mes remerciements vont aussi à M. Pierre Bourque et M. Pierre Gingras qui ont facilité grandement mon adhésion à l'ÉTS. Je tiens, aussi, à remercier les membres de jury pour l'honneur qu'ils me font en acceptant de participer à l'évaluation de ce mémoire.

Enfin, je tiens à remercier mes parents et toute ma famille pour leur encouragement. Qu'ils trouvent ici l'expression de toute ma reconnaissance.

TABLE DES MATIÈRES

Page

LISTE DES TABLEAUX

Page

LISTE DES FIGURES

Page

LISTE DES ABRÉVIATIONS, SIGLES ET ACRONYMES

ACDP	Architecture Cascadée dans le Domaine Pixel
ACDD	Architecture Cascadée dans le Domaine DCT
AMMS	Adaptive Motion Mode Selection
API	Application Programming Interface
CABAC	Context-based Adaptive Binary Arithmetic Coding
CAVLC	Context Adaptive Variable Length Coding
CF	Changement de format
CIF	Common Intermediate Format
CM	Compensation de Mouvement
CM-DCT	Compensation de Mouvement dans le domaine DCT
dB	Décibel
DBC	Débit Binaire Constant
DBV	Débit Binaire Variable
DCT	Discrete Cosine Transform
DLV	Décodage à Longueur Variable
DVD	Digital Versatile Disc
ELV	Encodage à Longueur Variable
EM	Estimation de Mouvement
FDVS	Forward Dominant Vector Selection
GOP	Group of Pictures
HDTV	High Definition TV
IDCT	Inverse DCT
IL	Insertion de Logo
IQ	Inverse Quantization
ISO	International Organization for Standardization
ITU	International Telecommunication Unit
MB	Macroblock
MPEG	Moving picture Experts Group

MC	Motion Compensation
MC-DCT	Motion Compensation in the DCT domain
ME	Motion Estimation
MSE	Mean Square Error
MV	Motion Vector
PSNR	Peak Signal to Noise Ratio
Q	Quantization
QCIF	Quarter Common Intermediate Format
QP	Quantization Parameter
RDB	Réduction du Débit Binaire
RDO	Rate Distortion Optimization
RRS	Réduction de la Résolution Spatiale
RRT	Réduction de la Résolution Temporelle
SAD	Sum of Absolute Differences
SDTV	Standard Definition TV
SVC	Scalable Video Coding
TCDP	Transcodeur Cascadé dans le Domaine Pixel
TCDD	Transcodeur Cascadé dans le Domaine DCT
TSDP	Transcodeur Simplifié dans le Domaine Pixel
TSDD	Transcodeur Simplifié dans le Domaine DCT
VLC	Variable Length Coding
VLC	Variable Length Decoding
VM	Vecteur de Mouvement
VQEG	Video Quality Experts Group

INTRODUCTION

La vidéo est un média incontournable pour transmettre de l'information ou des messages à l'audience. Elle est devenue de plus en plus populaire dans les dernières décennies surtout avec la numérisation, qui a permis le rapprochement entre le monde informatique et le monde électronique. La vidéo est utilisée presque partout dans la vie quotidienne : la télédiffusion (SDTV et HDTV), la TV sur les mobiles (*mobile TV*), surtout la lecture en continu sur Internet et réseaux mobiles (*le streaming vidéo*).

Une séquence vidéo brute et sans compression a un débit énorme de l'ordre de 180 Mb/s pour une séquence de traille 640 pixels x 480 pixels et 1 Gbps pour le HDTV. Cela représente un problème majeur pour la transmission ou le stockage des données. Ainsi, le codage (la compression) vidéo est utilisé pour réduire la taille d'une séquence (à typiquement 1,5 Mb/s pour MPEG-1, 4 ou 5 Mb/s pour MPEG-2), en se basant sur la redondance spatiale et temporelle. Après compression, la séquence vidéo est régie par un ensemble de caractéristiques qui sont généralement :

- La résolution spatiale (*spatial resolution*).
- La résolution temporelle ou le taux de trames par seconde (*frames per second fps*).
- Le débit binaire (*bitrate*).
- Le standard de compression (*compression standard*).

En général, le signal vidéo est compressé et stocké sans connaissance préalable des caractéristiques de l'appareil de l'usager ou des supports de transmission utilisés. Le transcodage vidéo est une technologie inévitable qui permet de transformer une séquence vidéo compressée en entrée, avec un ensemble de caractéristiques à une autre séquence vidéo compressée en sortie avec un autre ensemble de caractéristiques. La solution simple et directe pour implémenter un transcodeur vidéo consiste à décoder complètement la vidéo en entrée puis la réencoder avec les nouvelles exigences. Cette solution donne une qualité de la vidéo transcodée très proche de la vidéo en entrée, cependant, elle est très complexe en terme de calculs. Dans les dernières années, différentes architectures de transcodage vidéo ont été

réalisées afin de réduire cette complexité de calcul tout en essayant de maintenir une qualité similaire à la solution standard. Néanmoins, la plupart des architectures proposées se sont concentrées sur la réalisation d'une seule opération à la fois, soit la réduction du débit binaire, l'adaptation (*upscaling* or *downscaling*) de la résolution spatiale, l'adaptation de la résolution temporelle, le changement de format ou l'insertion de logo. Ceci mène à concevoir une architecture par cas d'utilisation, ce qui rend très fastidieux le développement et la maintenance du logiciel.

Le but de ce mémoire est de proposer un nouveau système unifié de transcodage vidéo pour effectuer une ou plusieurs opérations de transcodage de manière fiable et efficace, en bénéficiant des blocs communs entre les différentes opérations. Notre nouveau système permettra ainsi de surpasser les limitations des transcodeurs proposés dans la littérature en rajoutant plus d'opérations : l'insertion de logo et l'adaptation de format en supportant les standards les plus utilisés actuellement, soit MPEG-2, H.263, MPEG-4, H.264 et VC-1. Un tel système de transcodage permettrait, entre autres, d'avoir un transcodeur très rapide en calcul par rapport au transcodeur classique tout en maintenant une très haute qualité. En plus, les opérations de ce nouveau système seraient centralisées, ce qui réduirait les coûts de maintenance du logiciel et faciliterait le développement et l'intégration d'opérations futures.

Ce mémoire se divise en trois parties. Dans la première partie, nous présentons l'état de l'art du transcodage vidéo. Cette partie est composée de trois chapitres. Le chapitre 1 est une introduction générale au domaine du transcodage vidéo. Le chapitre 2 résume une analyse et discussion des algorithmes de transcodage vidéo dans le domaine spatial. Les transcodeurs présentés dans ce chapitre traitent les opérations de transcodage vidéo isolément. Pour compléter cet aspect, dans le chapitre 3 on trouve un grand nombre des travaux de recherche ayant essayé de proposer des systèmes pour réaliser plus d'une opération de transcodage vidéo.

La deuxième partie est consacrée à notre proposition pour effectuer plusieurs opérations de transcodage vidéo en un seul système. Le chapitre 4 présente notre nouveau système unifié

de transcodage vidéo capable d'effectuer une combinaison d'opérations de transcodage vidéo de manière très rapide et très fiable en qualité.

La troisième partie, est dédiée à l'environnement de simulations et aux tests réalisés. Dans le chapitre 5, nous parlons de l'environnement logiciel utilisé pour réaliser les simulations et des modifications qui y sont apportées. Le chapitre 6 illustre les simulations réalisées et les résultats obtenus.

À la fin, nous concluons ce mémoire par des recommandations et les perspectives futures de notre travail.

CHAPITRE 1

INTRODUCTION AU DOMAINE DU TRANSCODAGE VIDÉO

1.1 Introduction

Dans ce chapitre, nous présentons une introduction générale au domaine du transcodage vidéo. D'abord, nous présentons les principes généraux de la compression vidéo, puis nous définissons un transcodeur vidéo et nous exhibons l'utilité du transcodage. Ensuite, nous présentons les principaux domaines de transcodage vidéo dans lesquels nous pouvons réaliser les différentes applications. Après, nous spécifions les principales applications (ou cas d'utilisation) du transcodage vidéo qui sont implémentées dans les principaux domaines du transcodage vidéo. À la fin, nous concluons ce chapitre.

1.2 La compression vidéo

Le codage vidéo (compression vidéo) est basé sur un codage prédictif, c.-à-d. l'élimination des redondances temporelles entre les images (ou trames) consécutives ou l'élimination des redondances spatiales pour chaque image indépendamment des autres images ou les deux ensembles. Seule la différence entre l'image source et sa prédiction est compressée et envoyée dans le bit-stream (train de bits).

Dans la Figure 1.1, les trames I (*Intra Frames*) n'utilisent pas de prédiction temporelle et ainsi ne nécessitent pas le décodage d'autres trames pour être décodées. Les trames P (*Predictive Frames*) sont prédites à partir d'autres trames. Pour la trame P_2, l'erreur de prédiction entre la trame I_1 et la trame P_2, appelée résiduelle, est envoyée. Pour la trame P_3, la résiduelle entre la trame P_2 et P_3 est envoyée.

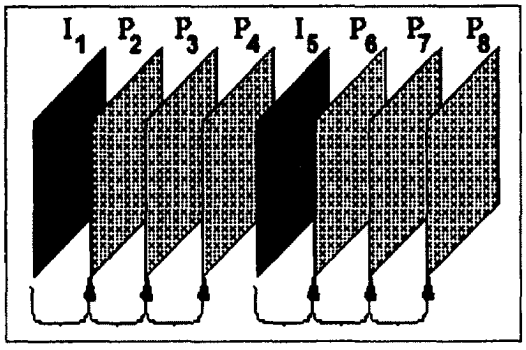

Figure 1.1 Une séquence vidéo avec des trames I et P.

Adaptée de Li (2004, p. 296)

Il existe une autre variante des trames P appelée trame B (*Bi-predicted Frames*) telle que représentée par la Figure 1.2. Une trame B, en général, fait appel à deux trames pour être décodée, contrairement aux trames P qui n'utilisent qu'une seule. La trame B_3 est encodée et décodée après P_2, mais elle est affichée avant P_2.

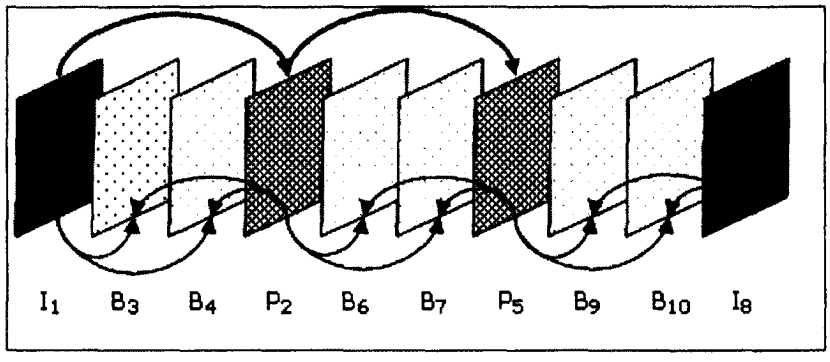

Figure 1.2 Une séquence vidéo avec des trames I, P et B.

Adaptée de Li (2004, p. 314)

La Figure 1.3 illustre le schéma simplifié d'un encodeur vidéo. Lorsqu'une trame I est encodée, elle est sujette à une transformée en cosinus discrète (*DCT*) et une quantification (*Q*). Le résultat de la quantification est encodé avec un codage arithmétique (*Variable Length Encoding*) et passe dans la boucle du décodeur où il est sujet à une quantification inverse (*Inverse Quantization*) puis une DCT inverse (*Inverse DCT*). À la fin, il est stocké dans la

mémoire-tampon (*Frame Store*) pour une utilisation future par les trames P ou B. Contrairement aux trames I, les trames P et B ne passent pas directement par la DCT, mais c'est la différence entre une trame P et son image prédite ou entre une trame B et ses images prédites qui passe par le processus d'encodage (DCT, Q puis VLC et la boucle de décodage). Lorsqu'une image P (ou B) est encodée, elle utilise les blocs d'estimation de mouvement (*Motion Estimation*) et compensation de mouvement (*Motion Compensation*) pour trouver la meilleure ressemblance (ou la plus petite erreur résiduelle) avec l'image précédente (ou les images précédentes dans le cas des trames B).

La DCT est responsable de décomposer le signal en fréquences. On distingue souvent les hautes fréquences et les basses fréquences. Les basses fréquences sont utilisées pour obtenir une approximation du signal en entrée alors que les hautes fréquences n'ajoutent que du détail à l'image et peuvent être ignorées ou codées plus agressivement sans perte notable de qualité visuelle.

La quantification utilise le signal issu de la DCT pour contrôler le niveau de compression. Cette étape permet d'avoir un gain important en compression mais cause une perte de qualité. En effet, plus le taux de compression sera élevé, plus la qualité diminuera.

Après la quantification, plusieurs coefficients DCT vont devenir nuls.

Le codage entropique (*VLC*) permet aussi de compresser le signal vidéo en plus de la quantification, mais sans causer de pertes.

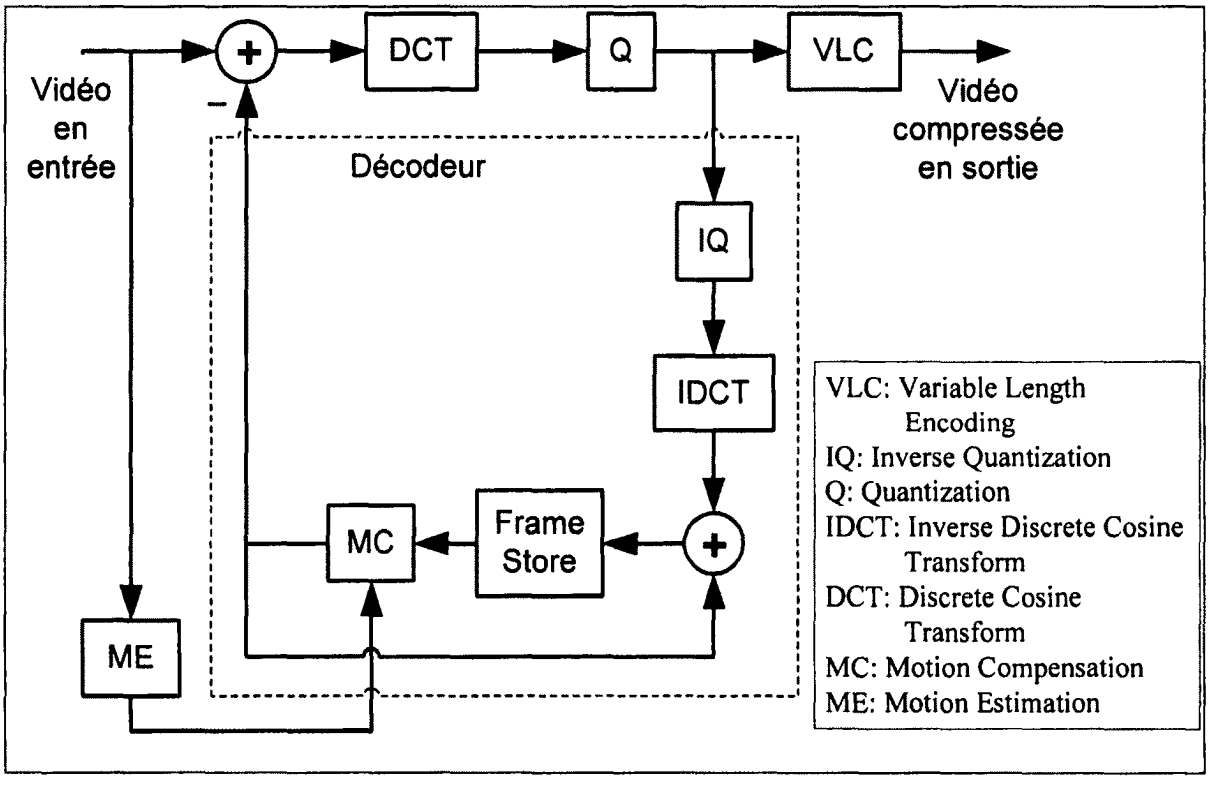

Figure 1.3 Schéma simplifié d'un encodeur vidéo.

Tirée de Xin (2005, p. 85)

La boucle de décodage est responsable de synchroniser le signal généré par l'encodeur dans la mémoire-tampon (*Frame Store*) avec celui décodé par le décodeur. En effet, dû à la perte causée par la quantification, l'image reconstruite sera différente de l'image originale. La quantification inverse (*IQ*) et la DCT inverse (*IDCT*) font l'inverse de la quantification et de la DCT respectivement. Étant donné que la DCT et la DCT inverse opèrent sur blocs de petites tailles, chaque trame est divisée en blocs de 16 pixels x 16 pixels, appelés macroblocs (MBs). La mémoire-tampon (*Frame Store*) est employée pour stocker les trames décodées pour une utilisation future. L'estimation de mouvement (*ME*) est l'étape la plus coûteuse en termes de complexité (temps de calcul) dans un encodeur. Cette opération produit des vecteurs de mouvements (*Motion Vectors*) pour chaque MB, qui sont envoyés avec le train de bits, pour indiquer la position spatiale de la meilleure prédiction qui a été utilisée. La compensation de mouvement (*MC*) est responsable de calculer la prédiction et de l'ajouter à la résiduelle pour reconstruire le MB encodé.

La Figure 1.4 illustre le schéma simplifié d'un décodeur vidéo. Ce qui est différent par rapport à la boucle de décodage incluse dans l'encodeur est le décodage entropique (*VLD*). Dans le cas du décodeur, les données ont été compressées avec un codage entropique (*VLC*). Le décodage entropique effectue l'opération inverse par rapport à ce dernier pour passer les données à la quantification inverse.

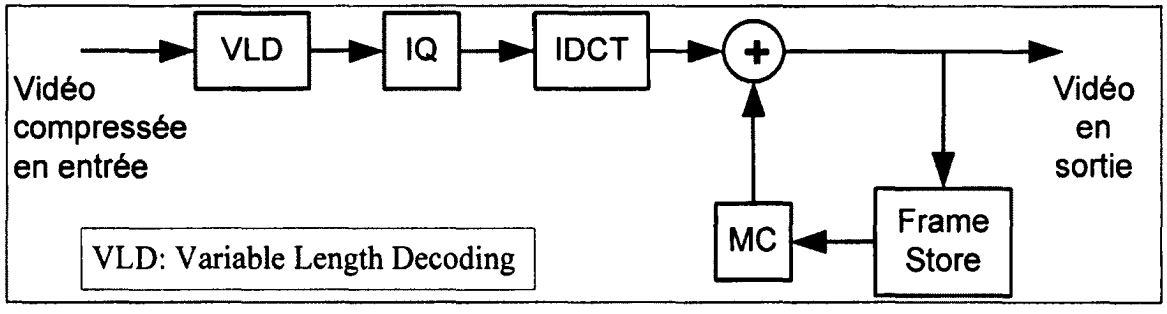

Figure 1.4 Schéma simplifié d'un décodeur vidéo.

1.3 Rôle d'un transcodeur vidéo

Dans le monde moderne, le marché du multimédia grandit d'une façon extrêmement rapide. La demande du contenu multimédia et surtout l'image et la vidéo est en pleine croissance. Pour accéder au contenu multimédia, qui est très varié par le fait de l'existence de plusieurs normes et formats [49], [54], les terminaux mobiles de toutes sortes telles que les ordinateurs portables, les assistants numériques personnels et les téléphones cellulaires sont devenus de plus en plus utilisés. Ces terminaux sont interconnectés entre eux par plusieurs réseaux hétérogènes filaires ou sans fils. Cependant, la diversité des terminaux mobiles et l'hétérogénéité des réseaux rendent difficile le processus de transport du contenu multimédia aux usagers finaux.

Le transcodage est une technologie clé pour pallier ce problème. Le transcodage vidéo est l'opération de convertir une séquence vidéo compressée d'un format en entrée à un autre format en sortie. Un ensemble de caractéristiques définit un format vidéo, telles que : le débit binaire, la résolution spatiale, la résolution temporelle, la syntaxe du train de bits. La Figure 1.5 illustre les rôles d'un transcodeur vidéo.

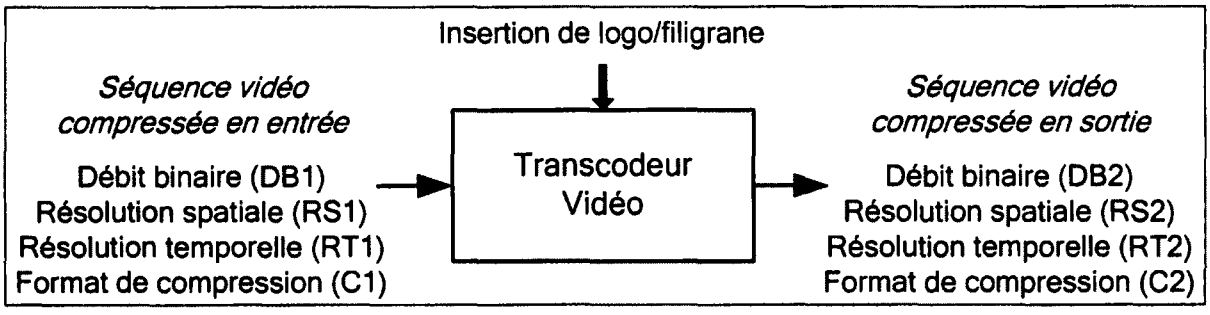

Figure 1.5 Rôles du transcodeur vidéo.

Tirée d'Ahmad (2005, p. 793)

Un transcodeur vidéo est mis au niveau d'un routeur, d'une passerelle, d'un serveur et de tout équipement intermédiaire de ce genre pour permettre l'interopérabilité et l'interconnexion entre le contenu diversifié, les différents réseaux hétérogènes filaires ou sans fils et les multiples terminaux fixes et mobiles [49].

Un transcodeur vidéo peut être utilisé dans différents cas d'utilisation [49]. Pour un cas, on peut utiliser le transcodeur pour réduire le débit binaire de la séquence vidéo imposé par les caractéristiques du réseau. Pour un autre cas, le transcodeur est employé pour réduire la résolution spatiale pour s'adapter aux capacités d'un téléphone cellulaire. Pour un troisième cas, le transcodeur est invité à changer le format entre H.263 et MPEG-4, et ajuster la résolution spatiale ainsi que la résolution temporelle pour satisfaire, aussi, les ressources limitées d'un téléphone cellulaire. De tels exemples de cas d'utilisation du transcodage vidéo sont présentés à la Figure 1.6.

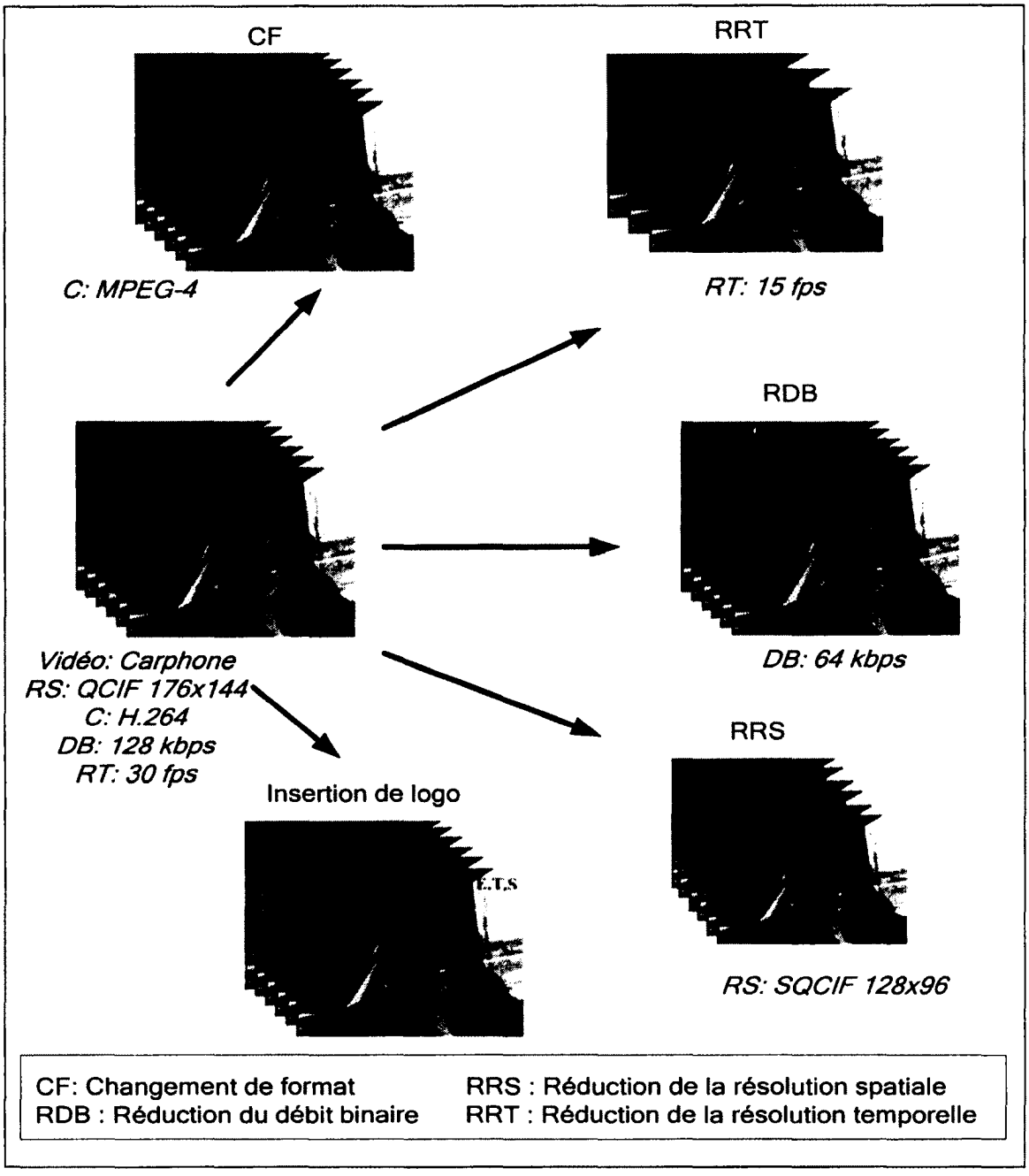

Figure 1.6 Cas d'utilisation du transcodage vidéo.

La méthode simple et directe pour réaliser un transcodeur vidéo consiste à cascader un décodeur et un encodeur telle qu'illustrée par la Figure 1.7. La séquence vidéo compressée en entrée est décodée. L'opération requise est réalisée selon les paramètres désirés en sortie puis la nouvelle séquence est réencodée avec ces nouveaux paramètres.

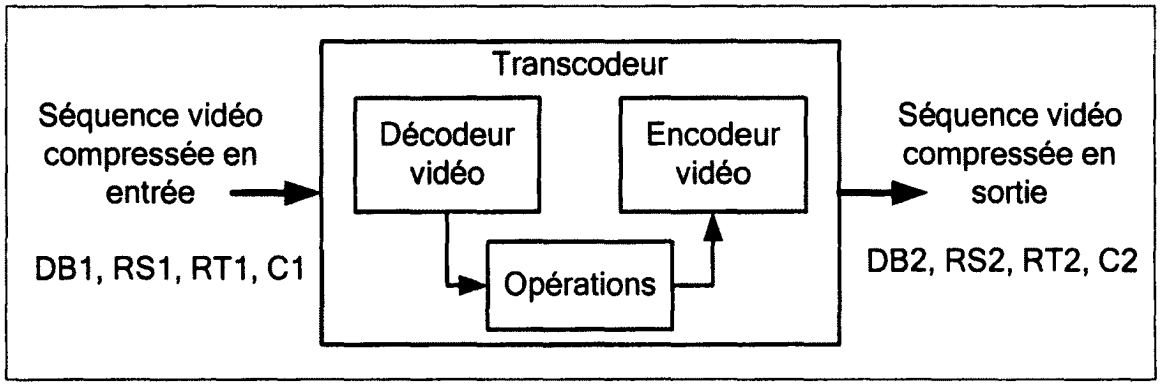

Figure 1.7 Schéma simplifié d'un transcodeur vidéo.

La Figure 1.8 montre le schéma détaillé d'un transcodeur vidéo. Le transcodeur réalise l'opération ou les opérations demandées de la manière suivante. Quand un flux vidéo arrive, il fait l'objet d'un décodage à longueur variable (VLD), une quantification inverse (IQ_1) puis une transformée en cosinus discrète inverse (IDCT) pour obtenir une trame intra ou une trame résiduelle. La trame résiduelle est sujette à une compensation de mouvement (MC) en utilisant les trames références stockées dans la mémoire-tampon. Après le décodage du flux vidéo, le transcodeur effectue l'opération ou les opérations demandées, ce qui donne une nouvelle séquence vidéo avec différents paramètres (débit binaire, résolution spatiale, résolution temporelle ou le standard de compression). Du côté de l'encodeur, une nouvelle estimation de mouvement (ME) est effectuée sur les trames décodées et qui ont été modifiées selon les paramètres désirés. Seules, la trame intra et l'erreur résiduelle obtenue après compensation de mouvement (MC), en utilisant la mémoire-tampon, subissent une transformée en cosinus discrète (DCT) puis une requantification (Q_2) puis sont envoyées après un encodage à longueur variable (VLC).

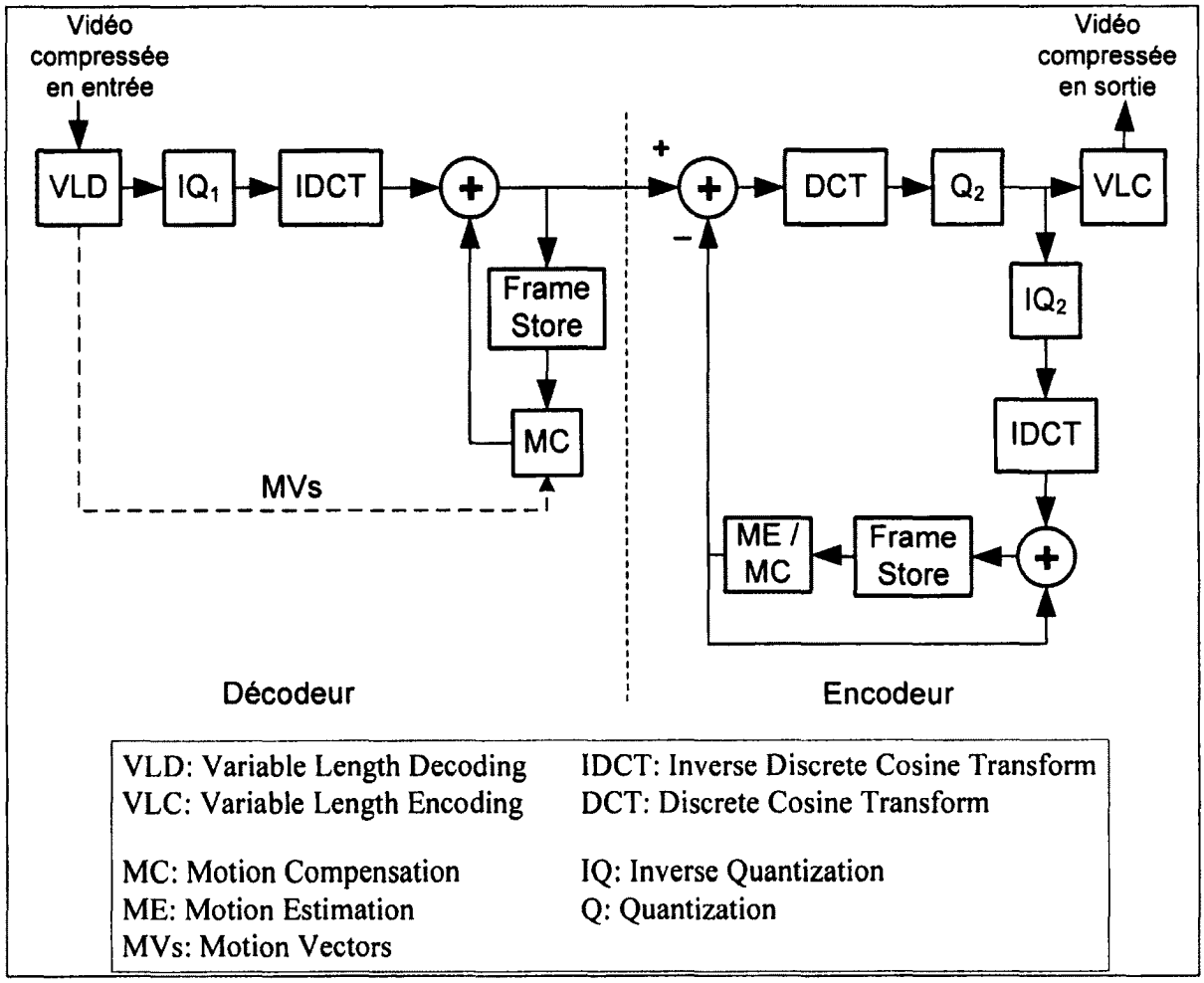

Figure 1.8 Schéma détaillé d'un transcodeur vidéo.

La recherche sur le transcodage vidéo est un domaine fascinant et très actif pour lequel un effort important a été consenti au cours des deux dernières décennies. La méthode simple et directe pour réaliser un transcodeur est de cascader un décodeur et un encodeur. Cependant, cette réalisation est très coûteuse en termes de calcul car le transcodeur ne réutilise pas les informations de la vidéo compressée disponibles au décodeur pour réencoder la vidéo. Ainsi, la majorité des travaux de recherche dans le domaine du transcodage vidéo focalisent sur la simplification de l'architecture cascadée tout en maintenant la meilleure qualité possible. Pour chaque application du transcodage, l'architecture utilisée est soit dans le domaine spatial (ou domaine pixel), soit dans le domaine transformé (ou domaine DCT :

Transformée en Cosinus Discrète ou *Discrete Cosine Transform*). Ces deux philosophies sont discutées dans la section suivante.

1.4 Architectures de transcodage dans les domaines spatial et transformé

Deux types d'architectures rapides ont été proposés dans la littérature pour effectuer une opération du transcodage vidéo. Le premier est dans le domaine spatial (pixel) et le deuxième est dans le domaine transformé (DCT) [60].

1.4.1 Architectures de transcodage dans le domaine spatial

Tel que mentionné, pour implémenter un transcodeur vidéo, une méthode simple et directe consiste à cascader un décodeur et un encodeur. Cette solution demande beaucoup de calculs car l'estimation de mouvement est refaite au complet, et de ce fait, elle n'est pas appropriée pour les transcodeurs temps réels. Pour pallier ce problème, une architecture rapide réutilise les métadonnées extraites du décodeur. Ainsi l'estimation de mouvement (ME) qui occupe à peu près 60-70% [42] du temps de transcodage est réduite voire éliminée. La Figure 1.9 représente un transcodeur vidéo rapide dans le domaine pixel (TCDP) pour la réduction du débit binaire. Toutes les opérations sont les mêmes que celles du transcodeur cascadé de la sous-section précédente à la différence que les vecteurs de mouvement décodés sont réutilisés au lieu d'être recalculés.

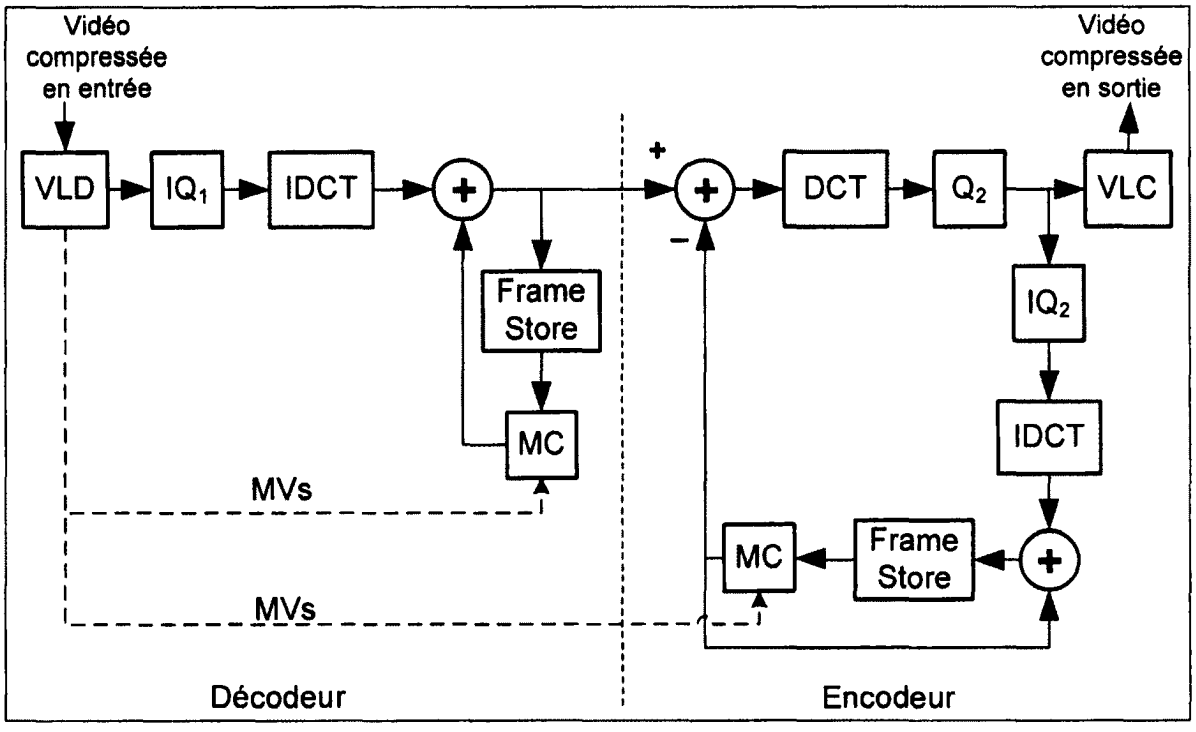

Figure 1.9 L'architecture de transcodage rapide dans le domaine pixel pour la réduction du débit binaire.

1.4.2 Architectures de transcodage dans le domaine transformé

Comme pour le domaine spatial, les architectures rapides de transcodage dans le domaine transformé partent de l'architecture cascadée d ont on tente de réduire la complexité en éliminant ou réduisant certains calculs. Une architecture rapide dans le domaine DCT pour la réduction du débit binaire [60] est illustrée dans la Figure 1.10.

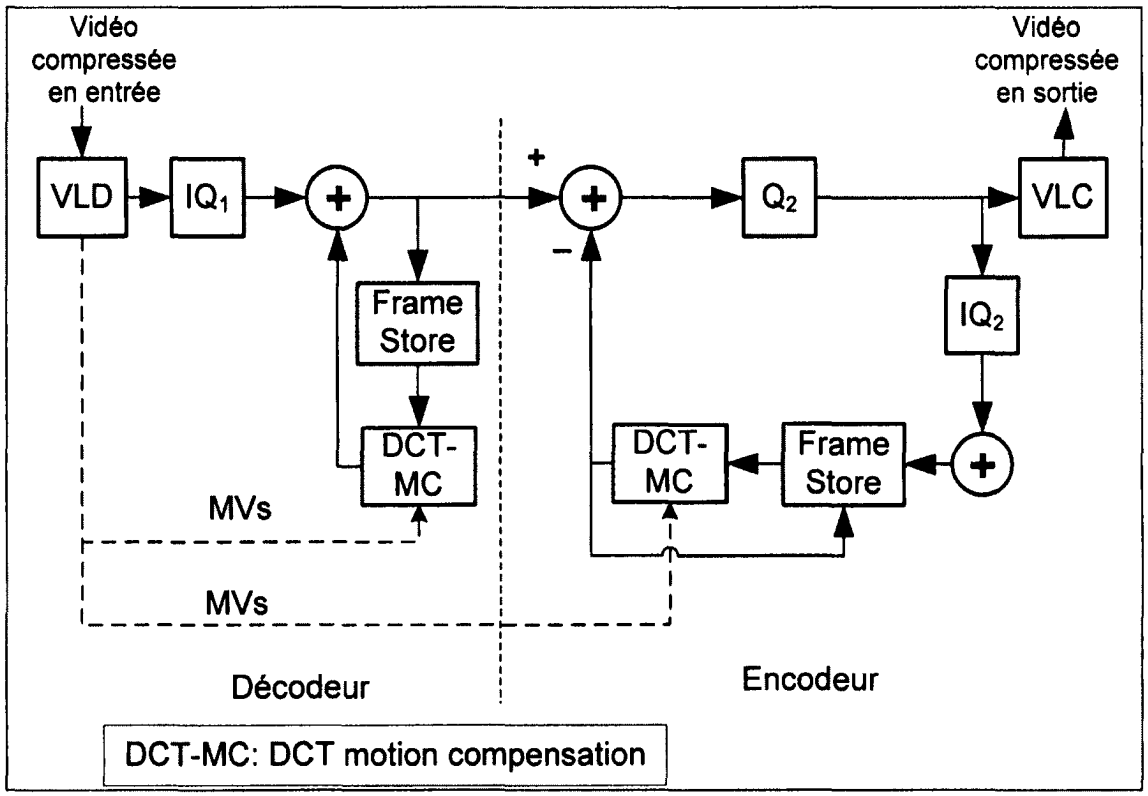

Figure 1.10 L'architecture rapide dans le domaine transformé pour la réduction du débit binaire.

Adaptée de Zhu (1998, p. 24)

Contrairement à l'architecture cascadée dans le domaine pixel (ACDP), l'architecture cascadée dans le domaine DCT (ACDD) opère directement sur les coefficients DCT. Du côté du décodeur, le train de bits en entrée est décodé avec un décodage à longueur variable (VLD) et une déquantification (IQ_1). La trame intra ou l'erreur résiduelle qui en résulte est dans le domaine transformé. L'erreur résiduelle est soumise à une compensation de mouvement dans le domaine DCT (DCT-MC) en utilisant les trames de références stockées dans la mémoire-tampon. De son côté, l'encodeur reçoit le flux vidéo décodé et le requantifie avec un paramètre de quantification Q_2, puis l'envoie après un à encodage longueur variable (VLC). Aucune estimation de mouvement (ME) n'est faite, mais une compensation de mouvement en réutilisant les métadonnées extraites du décodeur.

En comparant le domaine spatial avec le domaine transformé, ce dernier est moins complexe et plus faible en calculs [43; 60] car la transformée en cosinus discrète inverse (IDCT) du décodeur et la DCT/IDCT de l'encodeur sont éliminées[1]. En plus, la compensation de mouvement (MC) du décodeur et l'estimation de mouvement/compensation de mouvement (ME/MC) de l'encodeur sont réalisées en domaine transformé au lieu du domaine pixel. Cependant, ce gain est basé généralement sur la supposition que les DCT/IDCT sont des transformations linéaires[2] [44; 54]. Cette supposition n'est pas vraie tout le temps, à cause des différentes opérations d'arrondissement et de coupures effectuées par le décodeur et l'encodeur [54], ce qui cause une dégradation de la qualité de la vidéo. Le domaine spatial pallie ce problème en décodant complètement la vidéo pour réaliser le besoin demandé. Néanmoins, la haute qualité obtenue est au détriment de la complexité du transcodeur. Ainsi, la majorité des travaux se concentrent sur la minimisation du nombre de calculs et l'utilisation intelligente des informations récupérées du décodeur.

1.5 Fonctionnalités d'un transcodeur vidéo

Un transcodeur vidéo peut être utilisé dans diverses applications [1] telle que représentée par la Figure 1.11

---

[1] Les gains sont toutefois beaucoup moins intéressants pour les standards modernes comme H.264 et VC1.

[2] Une transformation linéaire est une transformation où : $T(\alpha p + \beta q) = \alpha T(p) + \beta T(q)$, avec α et β sont des scalaires; et p et q sont des fonctions (ou des variables ou constantes)

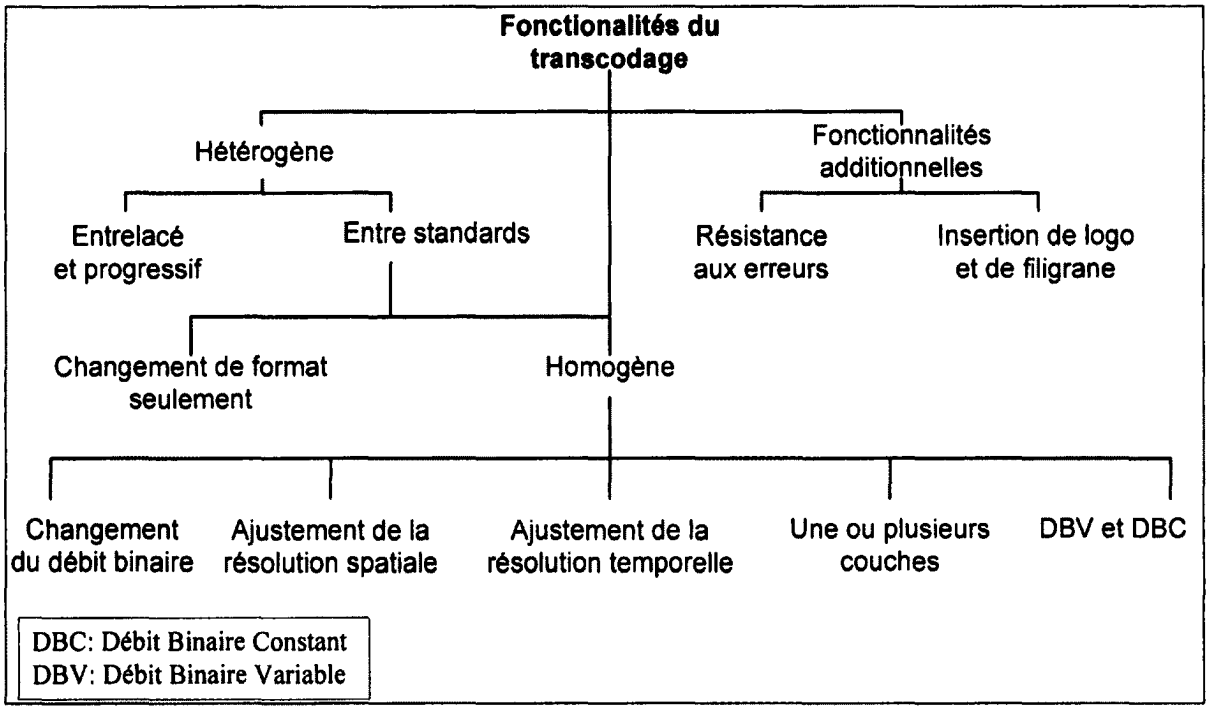

Figure 1.11 Fonctionnalités du transcodage.

Adaptée d'Ahmad (2005, p. 794)

Les applications du transcodage vidéo sont regroupées en trois grandes catégories : le transcodage homogène, le transcodage hétérogène et les applications additionnelles. Le transcodage hétérogène consiste à changer le standard de compression ou de passer du format progressif au format entrelacé ou vice versa. Le transcodage homogène englobe les opérations qui sont effectuées en utilisant le même format ou standard. Toutes les applications du transcodage homogène sont extensibles au transcodage hétérogène. Parmi ces opérations, on trouve :

- La réduction du débit binaire
- L'ajustement de la résolution spatiale
- L'ajustement de la résolution temporelle
- Le transcodage en utilisant plusieurs couches (*Scalable Video Coding*)
- Le transcodage avec débit binaire constant ou avec débit binaire variable.

De plus, les applications additionnelles qui forment la troisième catégorie sont généralement utilisées pour les deux premières catégories. On y trouve, par exemple, l'insertion de logos de télédiffuseurs.

1.5.1 Réduction du débit binaire

La réduction du débit binaire (RDB) vise à réduire le débit binaire imposé par un réseau. Le transcodeur vidéo doit implémenter cette opération tout en maintenant une faible complexité de calcul et une haute qualité de la vidéo autant que possible.

Cette application est requise dans des opérations telles que la lecture vidéo en transit (le *streaming vidéo*) ou la télédiffusion. Dans le domaine transformé, l'opération du transcodage vidéo est faite directement sur les coefficients DCT de la séquence vidéo en entrée. Le train de bits en entrée est sujet à un décodage à longueur variable (VLD) puis une déquantification (IQ_1). Les coefficients DCT qui en résultent subissent par la suite une requantification (Q_2). À ces coefficients obtenus après la requantification s'ajoutent toutes les métadonnées et les vecteurs de mouvements (VMs) obtenus après le VLD pour subir un encodage à longueur variable (VLC) et ainsi rencontrer le train de bits désiré en sortie. L'architecture ouverte (sans boucle de rétroaction) est illustrée dans la Figure 1.12 [36].

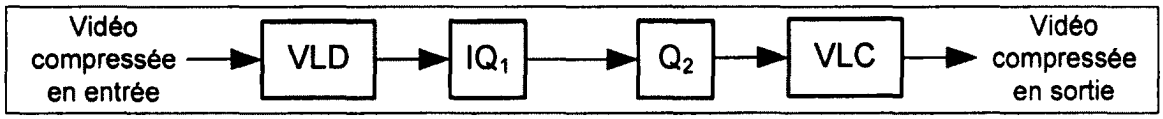

Figure 1.12 L'architecture ouverte de réduction du débit binaire.
Tirée de Nakajima (1995, p. 411)

Une deuxième variante de l'architecture ouverte est appelée transmission sélective [47]. Cette technique consiste à éliminer les coefficients DCT hautes fréquences puis à requantifier les coefficients DCT faibles fréquences avec le même paramètre de quantification ou un paramètre différent.

L'architecture ouverte est simple en réalisation et très rapide en calcul. Cependant, ses gains sont obtenus au coût d'une dégradation sévère de la qualité à cause de la désynchronisation entre l'encodeur et le décodeur appelée erreur de la dérive.

La cause de la dérive est la suivante. La compression de la vidéo est basée sur un codage prédictif. La trame courante est prédite d'une où plusieurs trames de références, et la différence (erreur résiduelle) entre la trame cible et les trames de références est codée seulement. Quand on coupe les coefficients hautes fréquences, pour que le décodeur puisse reconstruire les trames compressées par l'encodeur, il faut qu'il y ait une synchronisation entre le décodeur et l'encodeur, c'est-à-dire que les trames de références utilisées par le décodeur soient les mêmes que celles qui soient utilisées par l'encodeur. Or, dans le cas de l'architecture ouverte, les erreurs résiduelles ont changé après l'élimination des coefficients hautes fréquences dû à la requantification avec un paramètre de quantification différent par rapport au premier paramètre de quantification employé par la vidéo en entrée. De ce fait, du côté du décodeur, les trames reconstruites sont différentes de celles qui sont encodées et avec le temps, cette différence grandit de plus en plus à travers le groupe d'images (*group of pictures* GOP), et le résultat est une dégradation grave des trames reconstruites; ce qui implique une dégradation sévère de la qualité de la vidéo.

Différentes architectures ont été proposées dans la littérature afin de résoudre ce problème. Parmi les solutions proposées, il y a l'architecture simplifiée dans le domaine pixel [2]. Dans cette architecture, une seule boucle de reconstruction est partagée entre le décodeur et l'encodeur en se basant sur l'hypothèse que la DCT, l'IDCT et la compensation de mouvement (MC) sont des opérations linéaires [44; 54]. Ainsi, une seule IDCT et une seule compensation de mouvement (MC) sont nécessaires. L'architecture simple dans le domaine pixel est représentée par la Figure 1.13.

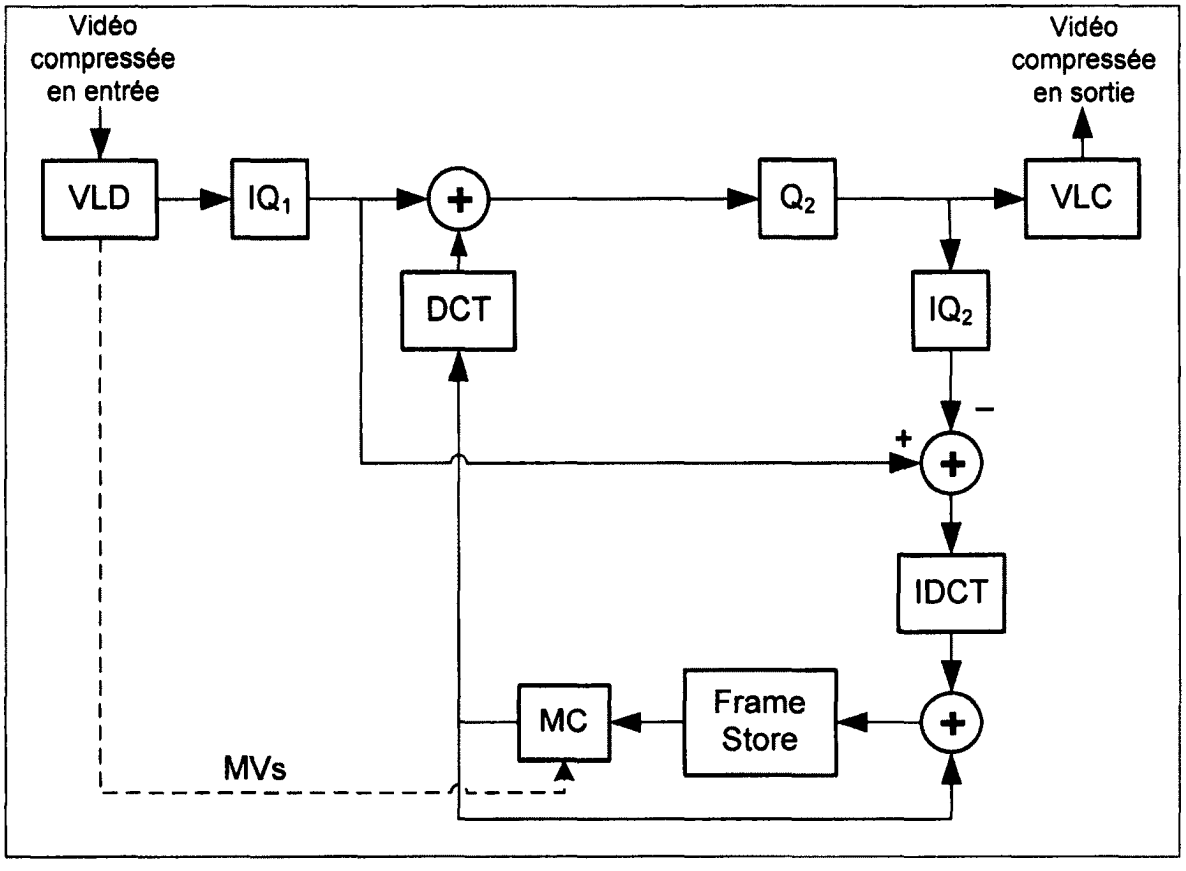

Figure 1.13 L'architecture simple de réduction de débit binaire dans le domaine pixel.
Tirée de Assuncao (1996, p. 2000)

1.5.2 Réduction de la résolution spatiale

Le but de la réduction de la résolution spatiale (RRS) est de réduire la résolution spatiale de la vidéo pour, par exemple, rencontrer les exigences d'un appareil mobile. L'architecture cascadée dans le domaine pixel (ACDP) est la manière simple et directe pour implémenter le transcodeur. Cette architecture consiste d'abord à décoder la vidéo en entrée complètement dans le domaine spatial, puis faire la réduction de la résolution spatiale dans le domaine pixel et finalement réencoder la vidéo avec résolution réduite pour rencontrer le train de bits désiré en sortie. La Figure 1.14 illustre cette architecture.

Dans cette architecture, le module STR (*Spatio-Temporal Reduction*) permet dans le cas de la réduction de la résolution spatiale de transformer un ensemble de 4 MBs en entrée à un seul

MB en sortie telle que décrit ultérieurement dans la sous-section 1.5.2.1. Le module MVCR (*Motion Vectors Composition and Refinement*) est responsable du mappage et de la composition des VMs en sortie à partir des VMs en entrée, puis le raffinement[3] des VMs finaux. Voir la sous-section 1.5.2.2.

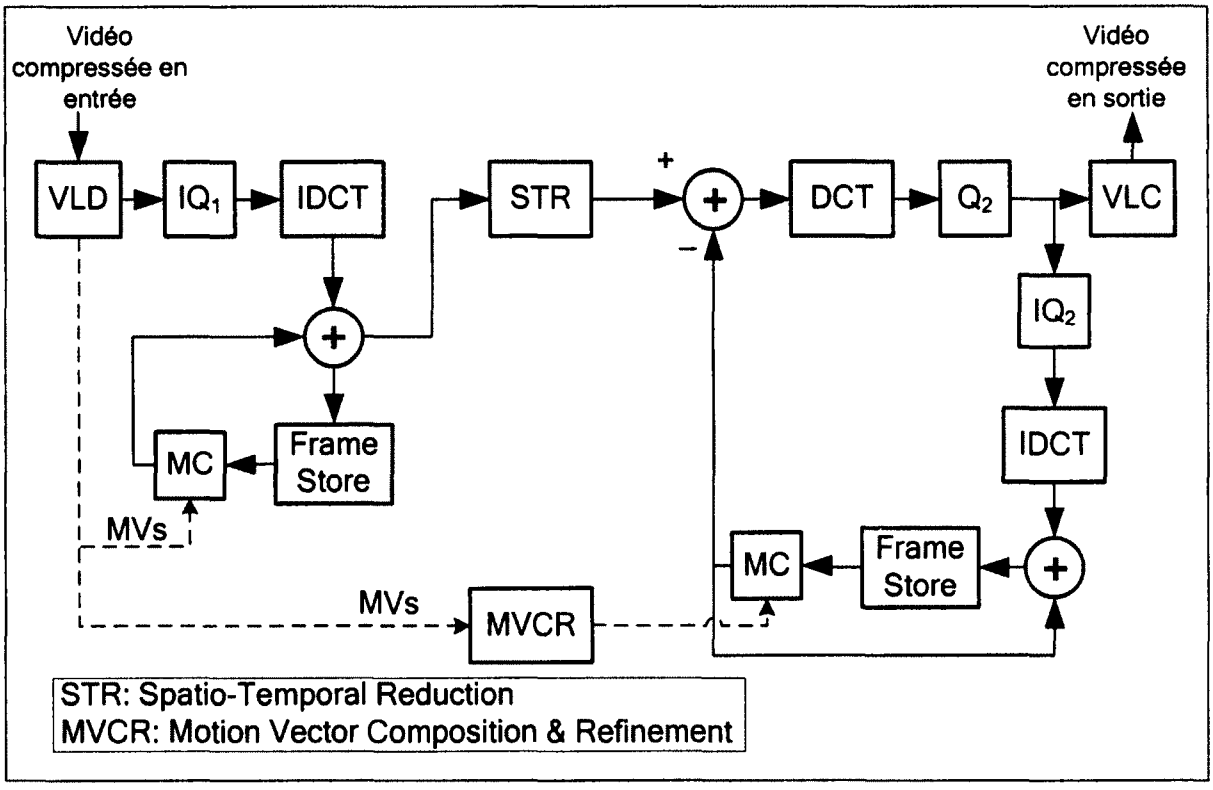

Figure 1.14 Réduction de la résolution spatio-temporelle par l'architecture cascadée dans le domaine spatial.

Tirée d'Ahmad (2005, p. 796)

[3] Le raffinement consiste à trouver un vecteur de mouvement optimal dans la région du vecteur de mouvement actuel. Le raffinement peut être plus ou moins complexe selon la zone de recherche.

Dans le domaine transformé, le transcodeur cascadé dans le domaine DCT (TCDD) [60] convertit la résolution de la vidéo en opérant directement sur les coefficients DCT. La compensation de mouvement dans le domaine DCT (DCT-MC) est l'opération la plus coûteuse en calculs dans ce domaine. Le TCDD [60] est illustré par la Figure 1.15.

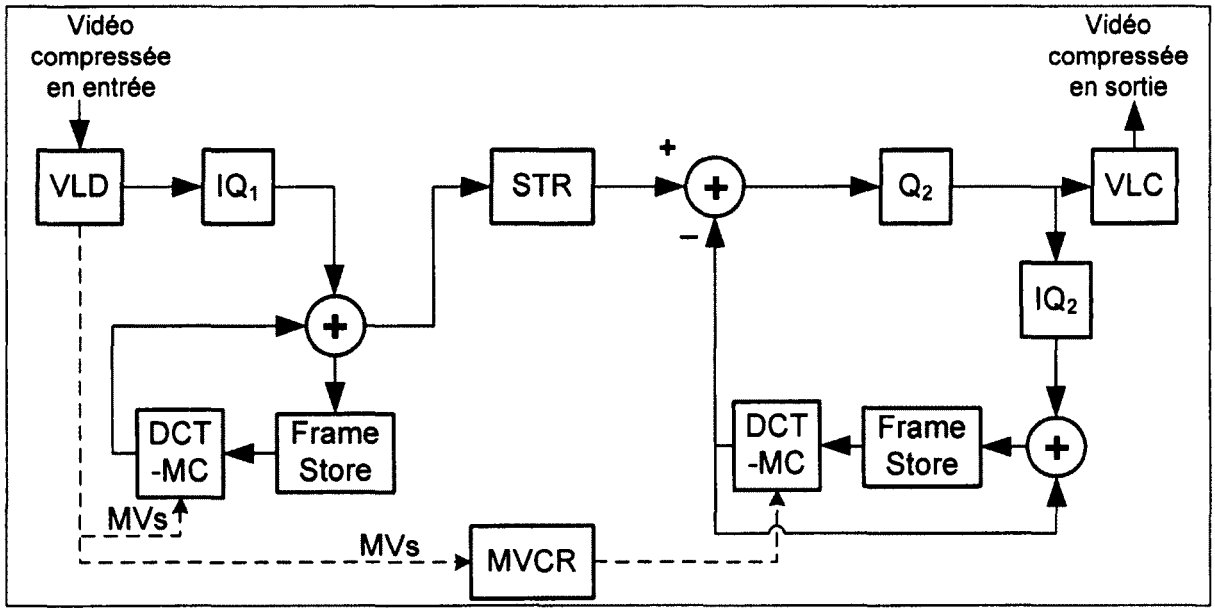

Figure 1.15 Réduction de la résolution spatio-temporelle par l'architecture cascadée dans le domaine transformé.

Tirée de Zhu (1998, p. 24)

Les travaux qui ont pour but de réduire la résolution spatiale de la vidéo focalisent soit sur la réutilisation efficace des informations extraites de la vidéo en entrée dans le contexte de l'ACDP [49], soit sur l'exploration de la possibilité de nouvelles architectures alternatives [49]. De ce fait, les architectures proposées sont dans le domaine spatial ou dans le domaine transformé. Cependant, les architectures proposées traitent généralement trois points représentés par les Figures 1.16 et 1.17 [1]: le ré-échantillonnage, le mappage des vecteurs de mouvements (VMs) et la décision du mode des macroblocks (MBs).

1.5.2.1 Méthodes de réduction spatiale des trames vidéo

La réduction de la résolution spatiale nécessite la mise à l'échelle de la taille des trames vidéos. Généralement, la réduction spatiale est effectuée par un facteur de deux, c.-à-d. passer de la taille de quatre MBs (2x2 MBs) en entrée à un seul MB en sortie. Ce principe est illustré par les Figures 1.16 et 1.17.

Pour réduire la taille des trames, différentes techniques [1] ont été proposées dans la littérature telles que :

- La moyenne [42] qui consiste à transformer chaque ensemble de m x m (généralement m = 2 pixels) à un pixel en utilisant la moyenne. Cette technique est simple à implémenter, mais la qualité résultante est souvent dégradée.
- Le filtrage et sous-échantillonnage : Shanableh et al. [42] ont proposé une technique pour réduire la taille des trames vidéo basée sur le filtrage et décimation. Après application d'un filtre passe-bas horizontalement et verticalement, la RRS est ensuite effectuée en décimant les pixels requis pour atteindre la résolution désirée.
- L'élimination des coefficients DCT hautes fréquences [42; 47] : Cette technique est utilisée dans les architectures de transcodage vidéo dans le domaine transformé où la manipulation des coefficients DCT est simple et directe.

1.5.2.2 Décision des modes de macroblocks (MBs)

En plus du rééchantillonnage des trames vidéo, les modes des MBs des trames vidéo requièrent quant à eux un mappage pour rencontrer les nouvelles caractéristiques de la vidéo réencodée. Plusieurs modes sont possibles et il importe de choisir celui offrant le meilleur compromis débit-distorsion (qualité par rapport au nombre de bits requis) qui est fonction du contenu visuel du MB à coder. La Figure 1.17 illustre le problème soulevé lors du mappage des modes des MBs.

La façon intuitive de réaliser ce mappage est décrite comme suit [1; 6; 47] :

- Coder le MB sortant comme skip si tous les MBs entrants étaient codés skip,
- Coder le MB sortant comme intra si au moins un des MBs entrants était codé intra,

- Coder le MB sortant comme skip s'il y a au moins un MB qui était codé inter et qu'aucun MB n'ait été codé intra.

1.5.2.3 Mappage des vecteurs de mouvements (VMs)

Lors du passage d'une haute résolution spatiale à une basse résolution spatiale, les VMs ont besoin d'être réestimés pour accommoder la nouvelle résolution. La Figure 1.16 illustre le problème du mappage des VMs. Ce mappage est effectué via différentes méthodes [1] :

- Aléatoire : Cette méthode consiste à choisir un VM d'une façon aléatoire [6]. Cette méthode est très rapide mais les résultats sont incertains.
- Moyenne : Cette méthode consiste à calculer le VM dans la taille réduite par la moyenne des VMs de la taille originale [42].
- Médiane pondérée : Cette méthode donne la meilleure qualité, mais requiert plus de calculs pour déterminer la médiane des VMs [42].

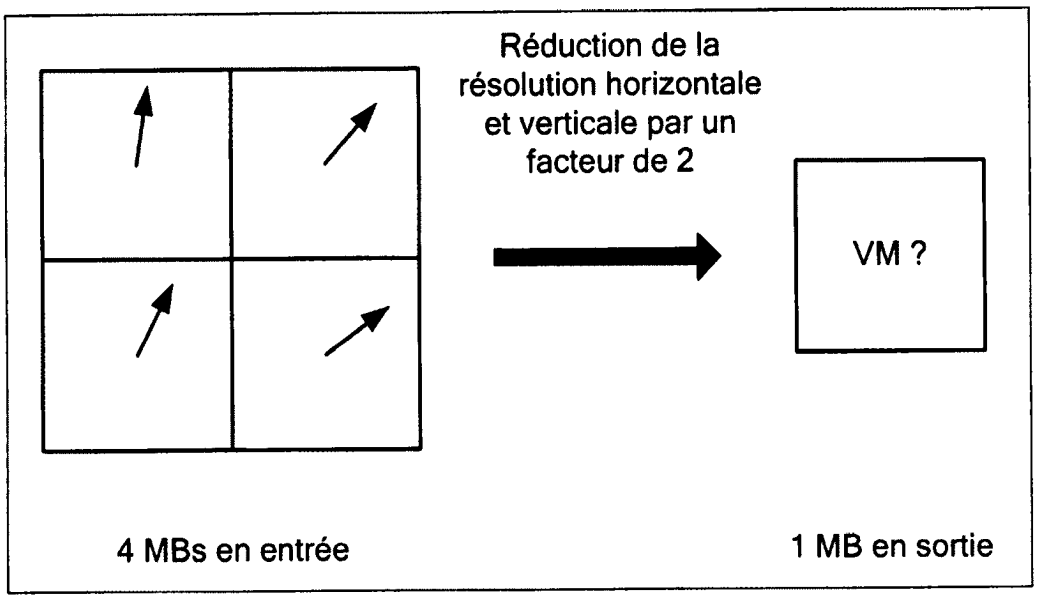

Figure 1.16 Composition des VMs dans la RRS par un facteur de 2.

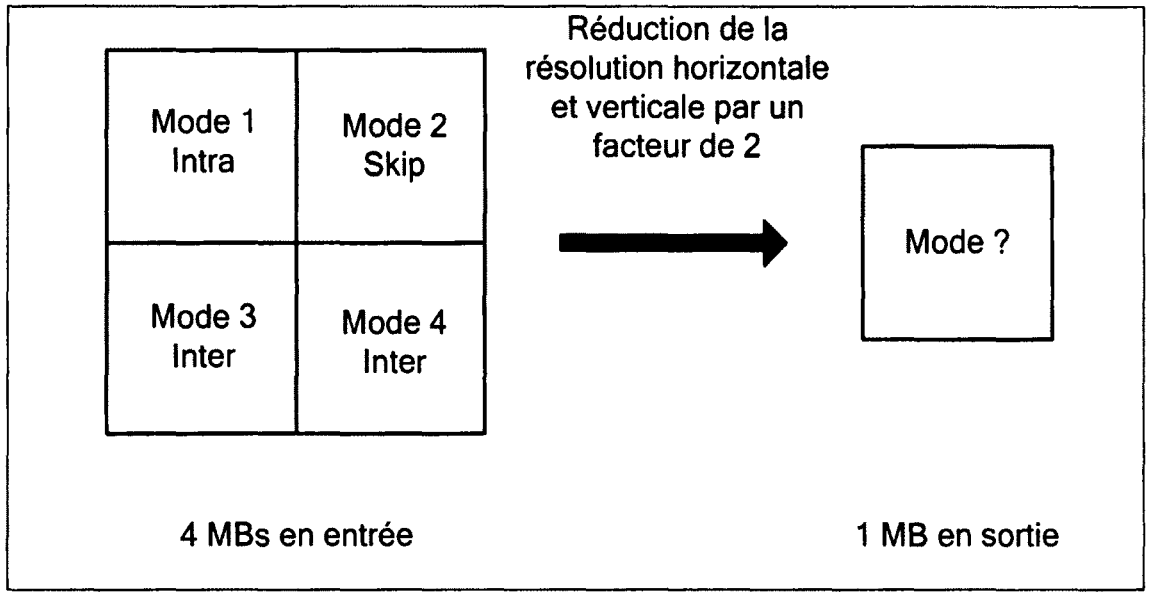

Figure 1.17 Sélection du mode du MB en sortie dans la RRS par un facteur de 2.

1.5.3 Réduction de la résolution temporelle

Le but de la réduction de la résolution temporelle (RRT) est de réduire le nombre de trames de la vidéo pour rencontrer les exigences d'un réseau ou les limites de calcul d'un appareil mobile. De même que pour la RRS, le TCDP est la méthode directe pour implémenter le transcodeur. Cette architecture est montrée dans la Figure 1.14 (p. 36). Dans cette architecture, le module STR (*Spatio-Temporal Reduction*) permet dans le cas de la réduction de la résolution temporelle de réduire le nombre des trames en entrée pour rencontrer les exigences de sortie. Le module MVCR (*Motion Vector Composition and Refinement*) est responsable du mappage et de la composition des VMs en sortie à partir des VMs en entrée, puis le raffinement des VMs finaux. Voir Figure 1.14.

Dans le domaine transformé, on retrouve aussi le transcodeur cascadé dans le domaine DCT (TCDD) comme l'architecture simple et directe pour la réalisation du transcodeur. La Figure 1.15 illustre cette architecture.

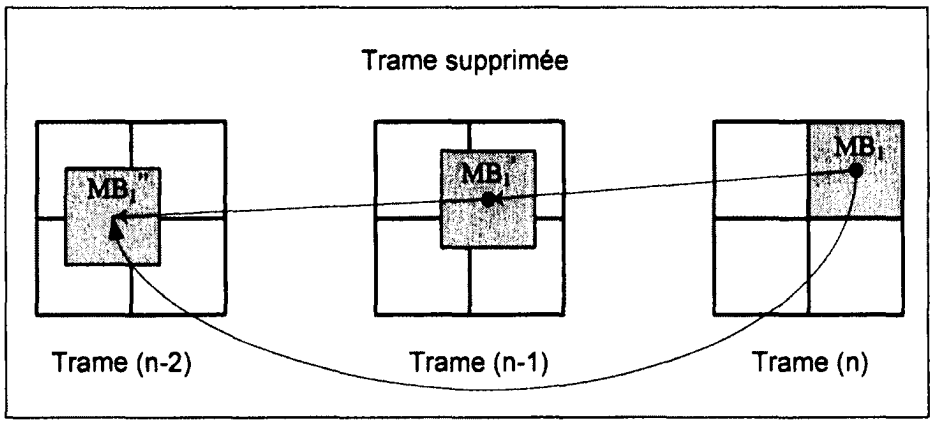

Figure 1.18 Problème de composition des VMs dans la RRT.

En supprimant des trames, une ré-estimation des vecteurs de mouvements (VMs) de la trame courante à partir de la trame précédente non sautée s'avère nécessaire telle que le montre la Figure 1.18, car les VMs du train de bits en entrée pointent vers des VMs qui n'existent plus. Différentes méthodes de recompositions des VMs ont été proposées dans la littérature [1] telles que :

- Interpolation bilinéaire : l'interpolation bilinéaire introduite par Hwang [16] permet de calculer les VMs des trames non sautées à partir des trames sautées si tous les VMs des trames adjacentes sont connus. D'abord, l'interpolation bilinéaire effectue un traçage des VMs des trames non sautées à partir de celles qui ont été sautées pour centrer la fenêtre de recherche des nouveaux VMs et ainsi réduire le nombre de calculs à effectuer. Ensuite, une adaptation des VMs résultant dans une fenêtre de recherche de taille réduite est réalisée.
- Composition du vecteur télescopique ou *Telescopic Vector Composition* (TVC) : l'algorithme TVC introduit par Shanableh [42] consiste à additionner tous les VMs de chaque trame non sautée jusqu'à la prochaine trame non sautée. Pour utiliser les VMs des trames sautées, cet algorithme nécessite la sauvegarde temporaire de toutes les trames d'un groupe d'images (*group of pictures* GOP).
- Sélection du vecteur dominant ou *Forward Dominant Vector Selection* (FDVS) : Youn et al. [23] ont proposé un algorithme Forward Dominant Vector Selection pour la ré-

estimation des VMs des trames non sautées. Cet algorithme sélectionne le VM du MB dominant dans chaque trame sautée jusqu'à la prochaine trame non sautée et met à jour la liste de tous les VMs à l'aide d'une table. Cet algorithme sera discuté plus en détail dans le chapitre 2.

1.5.4 Insertion de logo/filigrane

Pour des raisons de sécurité ou de protection des droits d'auteur, il est nécessaire d'insérer un filigrane ou un logo (filigrane visible) dans la vidéo compressée.
L'insertion de logo/filigrane (IL) affecte seulement une partie de la trame. Ainsi, on peut réutiliser les VMs entrants pour la partie non affectée par l'insertion du logo. Pour la zone où le logo est inséré, il faut prendre en considération [37]:

- La minimisation du nombre de MBs affectés en choisissant le meilleur endroit pour insérer le logo.
- Comment raffiner les VMs entrants, réencoder le mode des MBs et l'erreur résiduelle de telle sorte que ces traitements entraînent une complexité minimale et une dégradation minimale de la qualité.

Dans le domaine pixel, une fonction générale pour insérer un filigrane est définie dans [33] [39], et [58] comme suit :

$$W_n = \alpha_n \cdot L_n + \beta_n \cdot I_n \tag{1.1}$$

où L_n, I_n et W_n représentent le logo, la trame décodée et la trame avec logo inséré, respectivement. α_n et β_n représentent des facteurs scalaires qui contrôlent l'intensité du logo afin de donner une visibilité uniforme [58] et [34]. Le TCDP qui implémente cette fonction est illustré par la Figure 1.19 suivante.

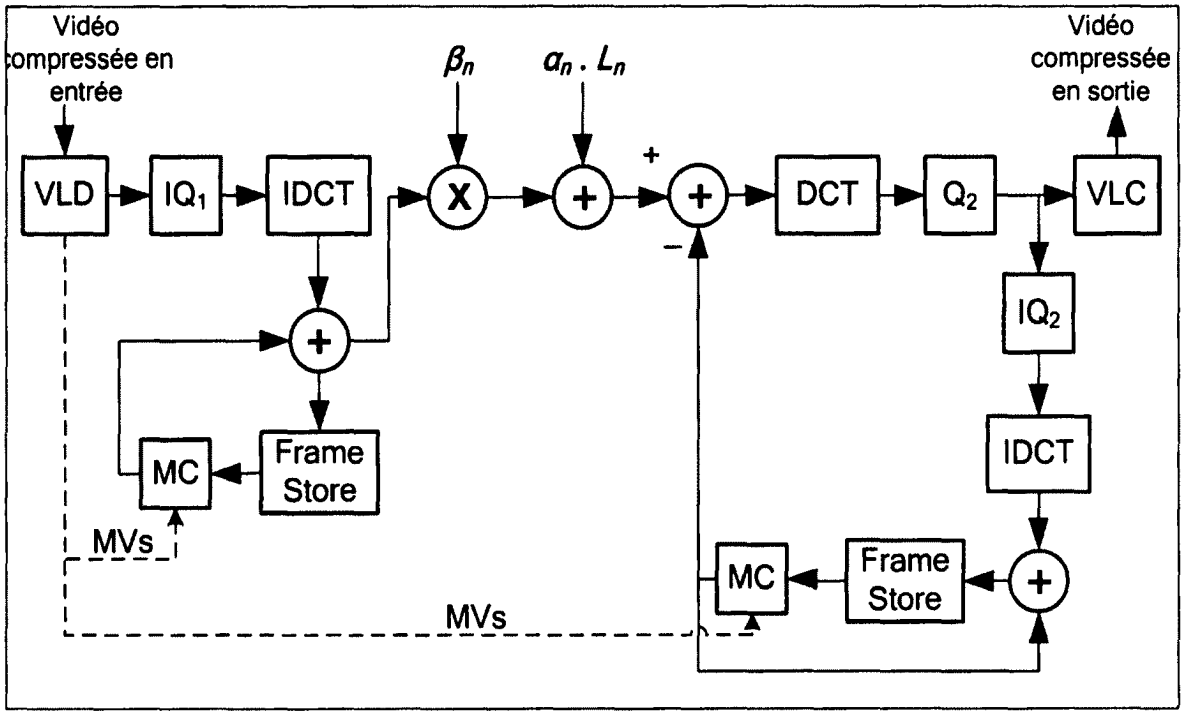

Figure 1.19 Insertion de logo/filigrane dans le domaine spatial.
Tirée de Youn (2000, p. IV-27)

En remplaçant β_n par 1- α_n, l'équation (1.1) se transforme en une fonction pour insérer un logo.

Dans le domaine DCT, différentes architectures ont été proposées dans la littérature pour insérer un logo. Une architecture simplifiée réalisant l'équation (1.1) est proposée par [58]. Cette architecture suppose que β est constant pour toutes les trames de la vidéo et est illustrée par la Figure 1.20. L'insertion de logo présentée à l'équation (1.1) étant, tout comme la DCT, une opération linéaire, sa complexité de calculs est la même qu'elle soit effectuée dans le domaine des pixels ou dans le domaine DCT (en prenant pour acquis que le logo est disponible dans le domaine dans lequel l'opération est effectuée).

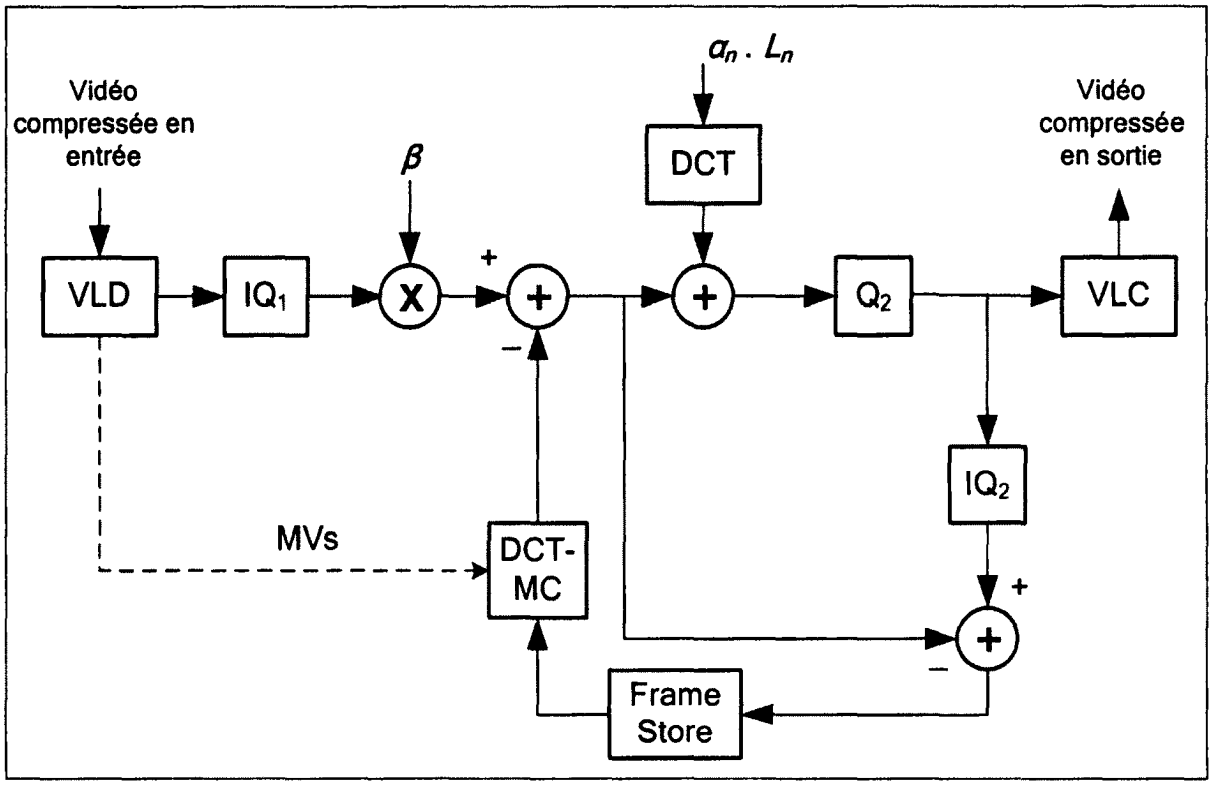

Figure 1.20 L'architecture simplifiée pour l'insertion de logo/filigrane dans le domaine transformé.

Tirée de Youn (2000, p. IV-28)

Un autre algorithme implémenté par un transcodeur cascadé dans le domaine DCT (TCDD) pour l'insertion de logo de format irrégulier est proposé dans [41].

1.5.5 Changement de format

Au cours des dernières années, plusieurs normes ont été développées, principalement par deux organismes internationaux : l'Union Internationale des Télécommunications (*International Telecommunication Unit*) ou ITU; et l'Organisation Internationale de Normalisation (*International Organization for Standardization*) ou ISO. L'organisme ITU a développé une série de normes : H.261, H.263 et H.263+/++ qui visent des applications vidéo à débits binaires réduits telles que la vidéo conférence et la vidéophonie [21]. De son côté, l'organisme ISO a créé MPEG-2 [19] pour les applications vidéo à haute qualité et à haut débit binaire telles que la télédiffusion et le DVD. MPEG-4 [20] a été développé par la

suite pour des applications comme la lecture vidéo en transit (le *streaming vidéo*) sur des appareils mobiles.

Le tout nouveau standard H.264/AVC [22] a été développé conjointement entre ISO/IEC et ITU-T VCEG pour le but d'atteindre une très haute compression. H.264/AVC peut produire la même qualité de vidéo avec un faible débit binaire d'une moyenne de 60% par rapport aux autres standards actuels [52].

Le tableau 1.1 [1] illustre les principales caractéristiques et propriétés des différents standards H.263, MPEG-2, MPEG-4 et H.264.

Tableau 1.1 Concepts clés de différents standards de compression vidéo
Adapté d'Ahmad (2005, p. 800)

Caractéristiques	**H.263+, H.263++**	**MPEG-2**	**MPEG-4**[4]	**H.264**
Profils	H.263 Annexes Support	Simple, Main, 4:2:2 SNR Scalable, Spatially Scalable, High	Simple, Simple Scalable, Core, Main, N-bit, Fine Granularity Scalable, etc	Constrained Baseline, Baseline, Main, Extended, High, High 10, High 4:2:2, High 4:4:4 Predictive, High Stereo, High 10 Intra, High 4:2:2 Intra, High 4:4:4 Intra, CAVLC 4:4:4 Intra
Approche de base	Basée sur le bloc	Basée sur le bloc	Basée sur le bloc	Basée sur le bloc
Unité de traitement	Macrobloc	Macrobloc	Macrobloc	Macrobloc

[4] MPEG-4 excluant le MPEG-4 AVC (*Advanced Video Coding*)

Caractéristiques	H.263+, H.263++	MPEG-2	MPEG-4[4]	H.264
Composantes des données	Luminance et chrominance	Luminance et chrominance	Luminance et chrominance	Luminance et chrominance
Types des trames	I, P, B, PB	I, P, B	I, P, B	I, P, B, SP, SI
Trames I	Moins (compression)	Plus (accès aléatoire)	Plus (accès aléatoire)	Moins (compression)
Codage entropique	VLC, SAC	VLC	VLC	UVLC ou CABAC
Précision de l'EM	½ pixel	½ pixel	Jusqu'à ¼ pixel	¼ pixel
Vecteurs de mouvement	Peuvent pointer à l'extérieur de la trame	Peuvent pointer à l'intérieur de la trame de réf. seulement	Peuvent pointer à l'extérieur de la trame	Peuvent pointer à l'extérieur de la trame
Transformation de bloc	8x8 DCT	8x8 DCT	8x8 DCT	4x4 entière
Multiples trames de références	Supporté par l'annexe U	Non supporté	Non supporté	Supporté
Taille du bloc de l'EM	8x8 ou 16x16	8x8 ou 16x16	8x8, 16x16, 16x8	16x16, 16x8, 8x16, 8x8, 8x4, 4x8, 4x4
Prédiction du bloc intra	Non	Non	Non	Oui
Formats supportés	Progressif	Progressif et entrelacé	Progressif et entrelacé	Progressif et entrelacé
Modes de prédictions	Trame seulement	Trame, image	Trame, image	Trame, image
Résolutions supportées	SQCIF, QCIF, CIF, 4CIF et 16CIF	De SQCIF à HDTV	Typiquement de SQCIF à `Studio`	De SQCIF à HDTV
Filtre de déblocage	Annexe J, frontière du bloc 8x8	Non	Oui, frontière du bloc 8x8, post filtrage	Oui, frontière du bloc 8x8, en boucle (in loop)

Un utilisateur final peut avoir un ou plusieurs équipements que ce soit un ordinateur de bureau, un ordinateur portable, un téléphone mobile, un assistant numérique personnel, une télévision, etc. Le changement de format permet l'accès universel au contenu multimédia des différents terminaux de l'usager final. Ainsi, le changement de format (CF) est requis pour transformer la vidéo de MPEG-2 utilisée par une chaîne de télévision à un client utilisant son téléphone cellulaire.

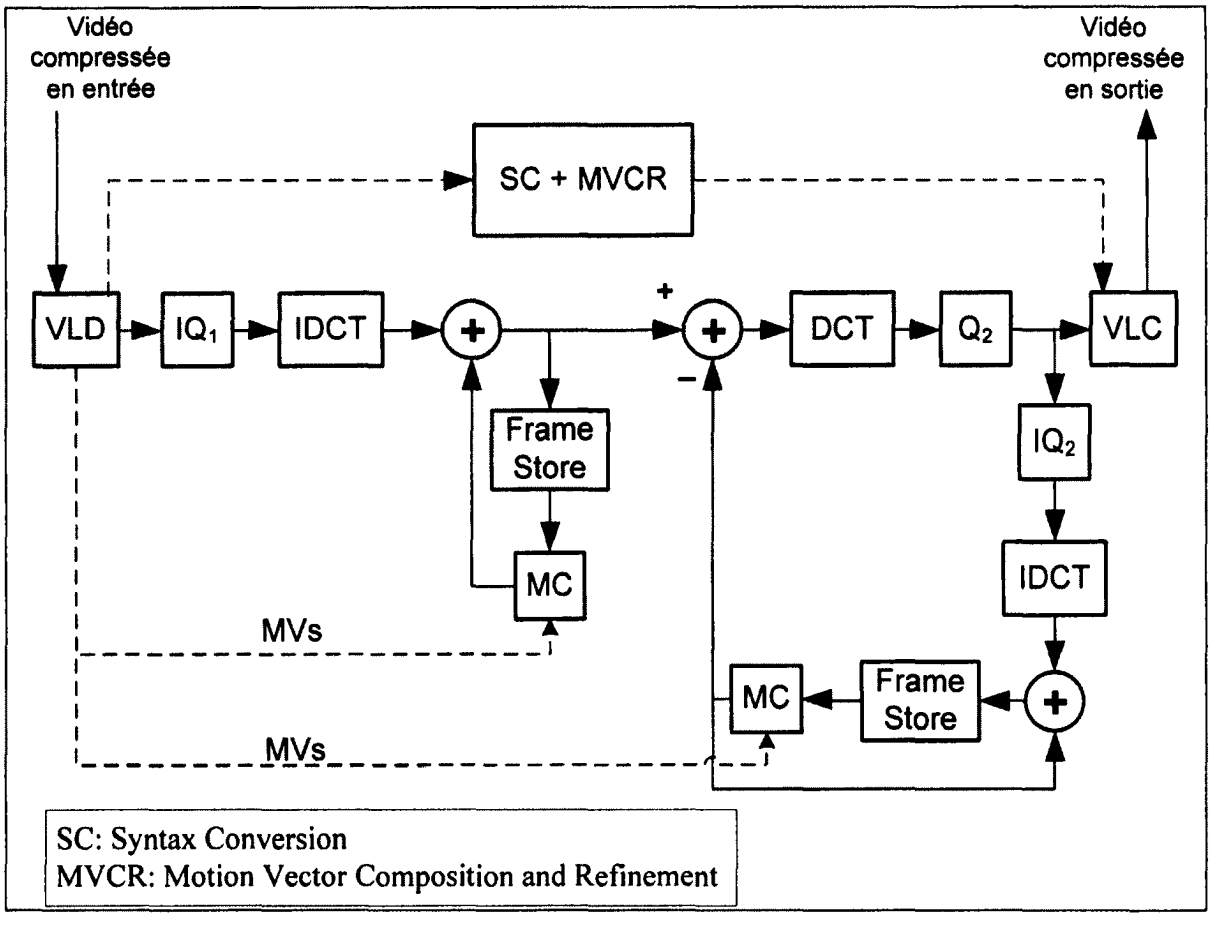

Figure 1.21 Architecture générique de CF dans le domaine spatial.
Tirée de Shanableh (2000, p. 102)

L'ACDP est la façon intuitive pour réaliser l'opération du CF. Cependant, cette architecture n'est pas générique étant donné les caractéristiques différentes entre les divers standards - MPEG-2 [15], MPEG-4[20], H.263 [21] et H.264 [22] - utilisés dans le marché du multimédia actuellement. Pour procéder au changement de format de MPEG-2 vers H.263,

Shanableh et al. [42] ont proposé un TCDP qui consiste à décoder le flux MPEG-2 entrant complètement dans le domaine spatial, puis réencoder ce flux avec la syntaxe de H.263. La Figure 1.21 illustre cette architecture [1; 42].

1.6 Conclusion

Nous avons présenté dans ce chapitre une introduction générale au domaine du transcodage vidéo. Nous avons commencé par une présentation du rôle du transcodage vidéo. Ensuite, nous avons présenté les deux principaux domaines des architectures de transcodage vidéo, soient le domaine pixel (aussi appelé domaine spatial) et le domaine DCT (ou le domaine transformé). Nous avons ensuite effectué un survol des principales applications du transcodage vidéo, les cas d'utilisation et les architectures simples proposées pour réaliser chaque opération du transcodage.

CHAPITRE 2

ANALYSE DES ALGORITHMES DE TRANSCODAGE DANS LE DOMAINE SPATIAL

2.1 Introduction

Que ce soit dans le domaine spatial ou dans le domaine transformé, différentes architectures et algorithmes de transcodage vidéo ont été réalisés dans le cadre de divers travaux de recherche pour implémenter chaque opération de transcodage vidéo : réduction du débit binaire, réduction de la résolution spatiale, réduction du nombre de trames par seconde, insertion de logo et changement de format. Ce chapitre présente une analyse plus détaillée des algorithmes de transcodage vidéo dans le domaine spatial pour les différentes applications.

Nous avons bâti notre sélection sur deux critères :

1. La flexibilité (domaine pixel vs. domaine DCT) : chaque algorithme sélectionné pour une fonctionnalité du transcodage vidéo doit être extensible pour d'autres fonctionnalités, d'où le choix des algorithmes dans le domaine spatial. Les transcodeurs dans le domaine DCT ont généralement une qualité dégradée par rapport aux transcodeurs dans le domaine pixel [54]. Aussi, si la DCT/DCT inverse est évitée dans le domaine DCT par rapport au domaine pixel, l'estimation de mouvement dans le domaine DCT est une opération très complexe par rapport au domaine pixel. En plus, avec le standard H.264, on ne peut pas réaliser efficacement un transcodeur générique dans le domaine transformé à cause du filtre de déblocage (*Deblocking filter*) utilisé pour améliorer la qualité de la vidéo de la trame décodée. [3; 32]. En plus, la compensation de mouvement est une opération très complexe à implémenter étant donné les différents modes de MBs utilisés dans H.264.
2. Le compromis qualité/performance : ce rapport est très important étant donné la diversité des réseaux de transport de la vidéo et du support de stockage et de traitement.

2.2 Réduction du débit binaire (RDB)

Plusieurs algorithmes ont été proposés dans la littérature pour la réduction du débit binaire d'une vidéo compressée. L'algorithme proposé par Bjork [6] est l'une des solutions qui donne le meilleur compromis qualité de la vidéo et temps de calcul nécessaire pour le transcodage. L'algorithme de Bjork [6] est basé sur une architecture cascadée dans le domaine pixel (ACDP) qui réutilise les méta-informations extraites du train de bits entrant pour minimiser le temps de traitement. La Figure 1.9 (p. 29) illustre cette architecture.
Après décodage du flux vidéo entrant, l'encodeur réencode les trames en requantifiant plus grossièrement le flux vidéo avec un paramètre de quantification $QP_2 \geq QP_1$. Les vecteurs de mouvements (VMs) demeurent inchangés et les modes sont réutilisés sauf pour les macroblocks (MBs) codés inter. Dans ce dernier cas, les modes seront réestimés de nouveau pour un meilleur contrôle du débit binaire et de la qualité.

2.3 Réduction de la résolution spatiale (RRS)

2.3.1 Principes généraux

La réduction de la résolution spatiale (RRS) peut être effectuée en utilisant un facteur entier ou arbitraire. Dans le cadre de notre projet, nous nous intéressons particulièrement à la réduction de la résolution par un facteur de deux pour minimiser le nombre de MBs impliqués dans la création de MBs de résolution réduite et réduire les calculs (voir Figures 1.14 et 1.15 p. 36-37). L'algorithme introduit par Bjork [6] pour la RRS est basé sur une ACDP flexible qui donne un niveau de qualité égal à l'architecture cascadée classique avec une réduction de la complexité du transcodeur. Cette méthode consiste d'abord à ré-échantillonner la trame réduite à partir de la trame originale en utilisant la méthode de la moyenne. Par la suite, les VMs sortants sont calculés à partir des VMs entrants en utilisant la médiane des VMs et ensuite une division par deux de ces VMs (réduction de l'échelle). Pour chaque ensemble de 4 VMs en entrée, un seul VM est donné en sortie; ce qui revient à un mappage de 4:1. Les VMs obtenus font l'objet d'un raffinement du côté de l'encodeur. À la fin, le mode des MBs est déterminé comme suit :

1. On code le nouveau MB comme intra s'il y a au moins un MB intra. Sinon, on code le nouveau MB comme inter s'il y a au moins un seul MB inter. Sinon, on code le nouveau MB comme skip.
2. On réévalue la décision des modes au niveau de l'encodeur. Par exemple, si l'énergie résiduelle dépasse un certain seuil, le mode pourrait changer.

Si Bjork et al. [6] ont proposé un mappage 4:1, Shanableh et al. [42] ont exploité davantage les caractéristiques des standards vidéo H.263 et MPEG-4 en proposant un mappage 1:1. Ainsi, chaque bloc 8x8 dans la taille réduite possède un VM qui correspond à un VM d'un MB 16x16 dans la taille originale. Les VMs dans la taille réduite sont générés par une mise à l'échelle par deux des VMs de la taille originale, puis un arrondissement au demi (½) pixel le plus proche. Finalement, un raffinement des VMs de ±½ pixel est effectué.

2.3.2 Réduction de la résolution spatiale dans H.264

À cause des différents modes de MBs employés en H.264 et la complexité de l'estimation de mouvement, l'opération de RRS se révèle plus complexe que les autres standards : H.263 et MPEG-2/4. Un MB dans une trame inter en H.264 peut être divisé en blocs 16x16, 16x8, 8x16 ou 8x8. En plus, chaque bloc 8x8 peut être subdivisé en sous blocs 8x8, 8x4, 4x8 ou 4x4. Chih-Hung et al. [8] ont développé un transcodeur pour la RRS en H.264. Cette architecture [8] est décrite comme suit.

2.3.2.1 Ré-estimation des VMs dans H.264

Pour ré-estimer les VMs, les 4 MBs en entrée sont divisés en blocs 8x8 avec un VM pour chaque bloc 8x8 tel qu'illustré par la Figure 2.1. De ce fait, nous allons avoir 16 blocs 8x8 avec 16 VMs. Ces 16 VMs sont transformés en 16 VMs dans le MB en sortie, c.-à-d., pour chaque bloc 4x4 un VM est attribué. Les 16 vecteurs de mouvement mv_i' sont numérotés tel qu'illustré à la Figure 2.1.

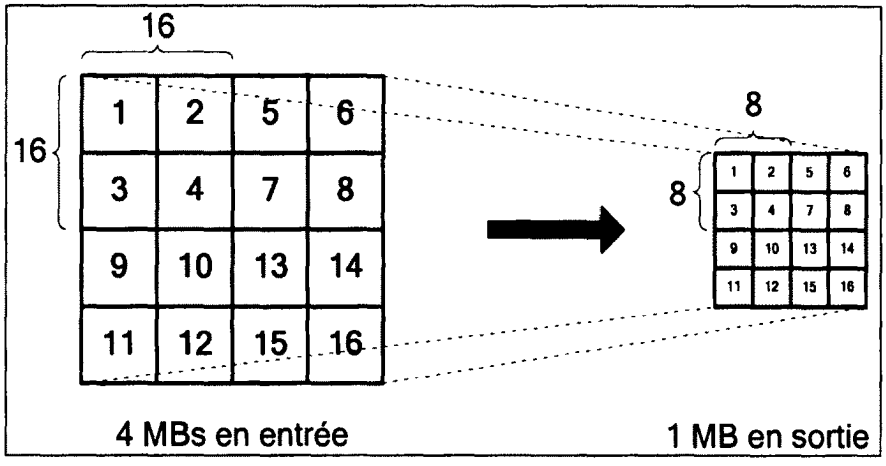

Figure 2.1 Ré-estimation des VMs pour la RRS par la méthode Chih-Hung.

Tableau 2.1 Fusion ascendante des VMs
Adapté de Chih-Hung (2004, p. 218)

Modes inter	Vecteurs de mouvements estimés
1 (inter16x16)	$mv_1'' = \text{Médiane } \{mv_1', mv_2', \ldots, mv_{16}'\}$
2 (inter16x8)	$mv_{21}'' = \text{Médiane } \{mv_1', mv_2', \ldots, mv_8'\}$ $mv_{22}'' = \text{Médiane } \{mv_9', mv_{10}', \ldots, mv_{16}'\}$
3 (inter8x16)	$mv_{31}'' = \text{Médiane } \{mv_i' \mid i=1, 2, 3, 4, 9, 10, 11, 12\}$ $mv_{32}'' = \text{Médiane } \{mv_i' \mid i=5, 6, 7, 8, 13, 14, 15, 16\}$
4 (inter8x8)	$mv_{4i}'' = \text{Médiane } \{mv_{4i-3}', mv_{4i-2}', mv_{4i-1}', mv_{4i}'\}$, i=1 à 4
5 (inter8x4)	$mv_{5i}'' = \text{Moyenne } \{mv_{2i-1}', mv_{2i}'\}$, i=1 à 8
6 (inter4x8)	$mv_{6i}'' = \text{Moyenne } \{mv_i', mv_{i+2}'\}$, i=1,2,5,6,9,10,13,14
7 (inter4x4)	$mv_{7i}'' = mv_i'$, i=1 à 16

Ensuite, à l'aide d'une méthode ascendante (bottom-up), les VMs de taille supérieure à 4x4, dans la taille réduite, sont calculés en combinant les VMs 4x4 (appelés mv_i' | i= 1..16) tel

qu'indiqué dans le tableau 2.1 et illustré par la Figure 2.2. Par exemple, pour avoir le VM pour inter16x16 (appelé mv_1''), la médiane des seize VMs 4x4 (mv_1', mv_2',..., mv_{16}') est calculée. Pour ré-estimer les deux VMs pour inter16x8 (mv_{21}'' et mv_{22}''), la médiane des huit VMs correspondants est calculée. Ainsi, les VMs correspondants pour chaque bloc sont calculés jusqu'aux blocs 4x4 où chaque VM 4x4 est pris directement ($mv_{7i}'' = mv_i'$).

Le but de cette combinaison est d'obtenir tous les VMs ressemblants aux VMs de la taille originale en utilisant une stratégie ascendante.

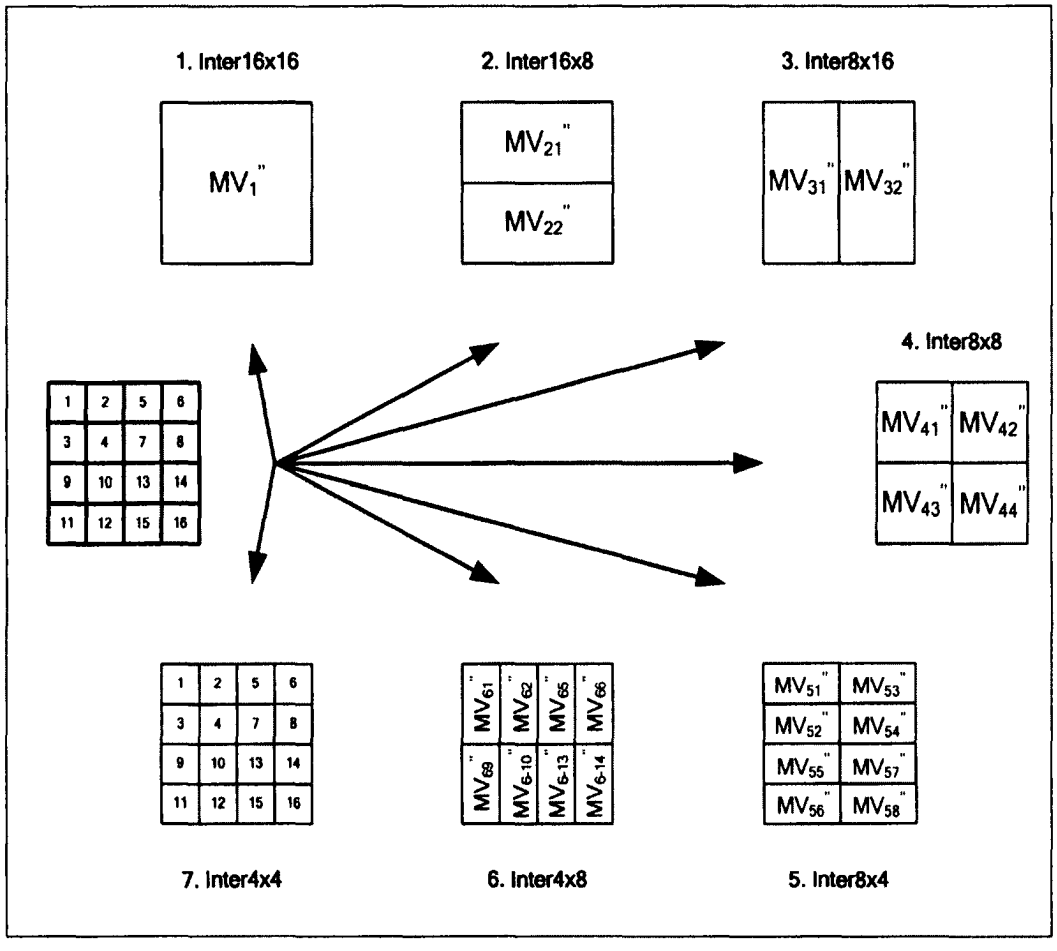

Figure 2.2 Fusion des VMs à partir des seize VMs des blocs 4x4 en utilisant la stratégie ascendante.

2.3.2.2 Décision des modes dans H.264

La décision des modes est effectuée en s'appuyant sur les informations extraites du train de bits en entrée pour minimiser le nombre de modes inter ou intra à vérifier et ainsi réduire les calculs. Étant donné que le mode skip a une faible complexité, il sera toujours testé.

Pour décider le mode de codage des MBs avec taille réduite, chaque bloc 8x8 d'un MB dans la haute résolution se voit attribuer un facteur 0, 1 ou 2 s'il est de type Inter, Intra4x4 ou Intra16x16 respectivement, tel qu'illustré par la Figure 2.3. De ce fait, en se basant sur la somme de ces facteurs dans un MB, nous pouvons déterminer si le nouveau MB est de type inter ou intra en commençant d'abord par tester l'éligibilité du type intra tel qu'illustré par le tableau 2.2. Par exemple, si le paramètre de quantification (QP) est inférieur ou égal à 35 (QP $\leq$ 35), intra4x4 sera testé si la somme des facteurs attribués à chaque MB (Figure 2.2) est supérieure ou égale à 48. Intra16x16 sera aussi testé si cette somme est supérieure ou égale à 48.

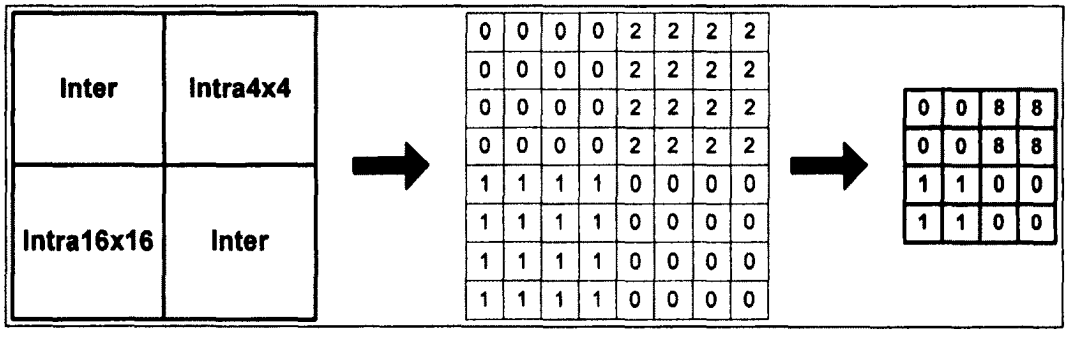

Figure 2.3 Facteur de mappage de la trame originale à la trame de taille réduite.

Tirée de Chih-Hung (2004, p. 219)

Tableau 2.2 Seuils de décision du mode intra
Tiré de Chih-Hung (2004, p. 220)

Limite du QP	Intra 4x4	Intra 16x16
QP ≤ 35	48	48
35 < QP ≤ 40	64	48
40 < QP ≤ 45	80	32
QP > 45	96	32

Dans le cas où le mode inter est testé, c'est-à-dire que pour un QP donné, la somme des facteurs est inférieure au seuil nécessaire tel qu'illustré par le tableau 2.2, Chih-Hung et al. proposent d'accélérer le processus de transcodage en testant certains modes inter seulement vu que H.264 en dispose d'un grand nombre de modes inter, de 16x16 jusqu'à 4x4. Ainsi, pour décider de tester un mode inter, la décision est effectuée si les sous blocs du MB ont des VMs inconsistants. L'opération de calcul de la quantité d'erreur est appliquée d'une manière ascendante entre les sous blocs des blocs 8x8, puis entre les 4 blocs 8x8 d'un MB de la taille réduite. La quantité d'erreur ($D_{i,j}$) est définie par l'équation 2.1 :

$$D_{i,j} = \left| mv_i - mv_j \right| < \left(Thr + \sqrt{QP_r - QP_o} \right) \; avec \; i,j = 1,2,3,4 \qquad (2.1)$$

où la quantité d'erreur $D_{i,j}$ égale la différence entre les vecteurs mv_i et mv_j. QP_r et QP_o sont les paramètres de quantification du train de bits en entrée et du train de bits en sortie respectivement, et *Thr* = *Thr4* pour les sous blocs d'un bloc 8x8 et *Thr* = *Thr8* pour les blocs d'un MB. Thr4 et Thr8 sont calculées par les équations 2.2 et 2.3 respectivement.

Pour les blocs 8x8, si la quantité d'erreur ($D_{i,j}$) est inférieure au seuil ($Thr8 + \sqrt{QP_r + QP_o}$), les modes inter 4x4, 4x8 et 8x4 sont exclus des calculs de la distorsion (RDO). De même, les modes inter 8x8, 8x16 et 16x8 sont exclus si la quantité d'erreur ($D_{i,j}$) est inférieure au seuil ($Thr4 + \sqrt{QP_r + QP_q}$). Thr4 est calculé par l'équation 2.2 de la manière suivante. D'abord, lors du calcul de la médiane pour 4 VMs, six valeurs relatives sont testées. Si les VMs d'une

paire sont différents, on rajoute 1 à N et on rajoute la différence à SumDifference. ScaleFactor est le ratio de réduction de résolution utilisé. Dans ce cas, il est ½.

$$Thr4 = 1 + \frac{SumDifference}{N \times ScaleFactor} \tag{2.2}$$

$$Thr8 = 0 \tag{2.3}$$

2.3.2.3 Raffinement adapté des VMs dans H.264

La fenêtre de recherche utilisée pour raffiner les VMs dans la méthode proposée par Chih-Hung est de l'ordre de ± 3 (± ¾ pixel).

2.4 Réduction de la résolution temporelle (RRT)

2.4.1 Principes généraux

Pour réaliser la réduction de la résolution temporelle (RRT), plusieurs algorithmes ont été implémentés dans la littérature. Dans ce qui suit, nous allons décrire la méthode *Forward Dominant Vector Selection* (FDVS) [23] qui donne une importante réduction du temps de calcul tout en maintenant une qualité approximativement égale à celle de l'architecture cascadée. La méthode proposée sélectionne un VM dominant à partir des quatre MBs adjacents. Un VM dominant est défini comme le VM utilisé par le MB dominant. Ce dernier est le MB qui a la plus grande partie de chevauchement avec le bloc pointé par le VM entrant (voir Figure 1.18).

Cette méthode utilise une seule table pour sauvegarder les VMs des MBs des trames sautées et aussi pour calculer et mettre à jour les VMs des MBs des trames non sautées. Au cas où des trames intra existeraient parmi les trames sautées, la méthode assume des VMs zéro et continue le processus de composition. Le calcul des VMs des trames non sautées est donné par l'équation 2.4 :

$$(Bx, By)_n = \left(\sum_{d=k}^{1} (Vx)_{n-d} + (Ix)_n, \sum_{d=k}^{1} (Vy)_{n-d} + (Iy)_n \right) \quad (2.4)$$

où $(Bx, By)_n$ représente le nouveau VM pour la trame courante, $(Vx, Vy)_{n-d}$ représente le VM dominant (le VM de la plus grande région) dans la trame (n-d) et (Ix, Iy) représente le VM en entrée de la trame courante.

Après le calcul des nouveaux VMs des trames non sautées à partir des trames sautées, un raffinement des VMs dans une fenêtre de recherche de très petite taille de l'ordre de ± 2 est effectué. En se basant sur ce raffinement, la décision du mode est ré-estimée.

2.4.2 Extension de la méthode FDVS à H.264

La méthode de Youn et al. [23] est surtout convenable pour les standards H.263, MPEG-2 ou MPEG-4 où chaque MB peut avoir une partition 16x16 ou quatre partitions 8x8. Qiang et al. [40] ont étendu la méthode FDVS à H.264. Cette extension, appelée *Adaptive Motion Mode Selection* (AMMS) est basée sur une architecture cascadée dans le domaine pixel (ACDP) qui applique les mêmes principes de FDVS mais en supportant les nouveaux modes de H.264. Elle est donc basée sur la division des MBs en bloc 8x8 ou en bloc 4x4, au cas où le mode utilisé en entrée est moins que 8x8, puis le traçage des VMs des trames sautées à partir des trames non sautées utilisant FDVS, et à la fin le calcul des nouveaux MBs en utilisant une technique appropriée au nouveau standard H.264.

2.5 Changement de format (CF)

Tel que déjà mentionné, différents standards vidéo ont été développés au cours des dernières années par deux organismes internationaux ISO (*International Organization for Standardization*) et ITU (*International Telecommunication Unit*). Les standards les plus utilisés dans le multimédia actuellement englobent notamment MPEG-2, MPEG-4, H.263 et H.264. Chaque standard a pour cible une catégorie d'application bien spécifique. MPEG-2 vise les DVD et la télédiffusion (définition standard SDTV ou haute définition HDTV) ayant une haute qualité [19]; H.263 est utilisé pour les applications multimédia à faible débit

comme la vidéophonie et la vidéo conférence [21]; et MPEG-4 est le cœur des applications telles que la lecture en continu [20].

Le tout nouveau standard de codage vidéo H.264 [22] surpasse de loin ces anciens standards. Il est conçu pour toutes sortes d'applications : vidéoconférence, diffusion, stockage et transmission. Les caractéristiques de H.264 telles que montrées par le tableau 1.1 sont :

- Utilisation de nouvelles méthodes de codage entropique : CAVLC qui donne une amélioration de 25% par rapport au codage entropique classique; et CABAC qui surpasse CAVLC de 10% mais avec plus de complexité.
- Utilisation d'une nouvelle transformation entière 4x4 au lieu de la DCT.
- Filtre de déblocage en boucle pour prévenir les artefacts.
- Prédiction spatiale pour les MBs intra en utilisant neuf prédictions directionnelles pour intra4x4 et quatre prédictions directionnelles pour intra16x16.
- Utilisation de plusieurs trames de références pour coder un MB allant jusqu'à 16 trames.
- Précision au quart de pixel rendant plus exacte la compensation de mouvement.
- Support de nouvelles tailles de bloc allant de 4x4 jusqu'à 16x16.

Conséquemment, nous avons divisé le changement de format (CF) en cinq parties.

- Le CF de MPEG-2, MPEG-4 et H.264 vers le nouveau standard H.264 : de plus en plus d'appareils supportent le standard H.264 et un transcodage de ses prédécesseurs se voit nécessaire.
- Le CF de H.264 vers les anciens standards MPEG-2, MPEG-4 et H.263 : une rétrocompatibilité est d'une grande importance pour les anciens appareils.
- Puis, entre les trois standards basés sur la DCT, c.-à-d. le CF entre MPEG-2 et MPEG-4, entre MPEG-2 et H.263 et entre H.263 et MPEG-4.

2.5.1 CF de MPEG-2, MPEG-4 et H.263 vers H.264

2.5.1.1 MPEG-4 à H.264

Lee et al. [26; 27; 28] ont proposé une méthode basée sur une ACDP avec réutilisation des méta-informations pour le changement de format de MPEG-4 à H.264. La nouvelle méthode est basée sur les statistiques (voir tableaux 2.3 et 2.4) où des expérimentations ont été effectuées sur plusieurs séquences vidéo tests pour connaître les probabilités de transformation des quatre modes de MBs utilisés par MPEG-4 (Inter16x16, Inter8x8, Intra16x16 ou Skip) vers les dix modes de MBs utilisés par H.264 (les huit modes inter : Inter16x16, Inter16x8, Inter8x16, Inter8x8, Inter8x4, Inter4x8, Inter4x4; les deux modes intra : Intra16x16, Intra4x4; ou le mode Skip). En analysant les tableaux 2.3 et 2.4, on peut remarquer que pour chaque mode MPEG-4 en entrée, certains modes de H.264 en sortie ont une probabilité plus élevée, et ce sont ces modes qui seront testés au niveau de l'encodeur H.264.

Tableau 2.3 Pourcentage de conversion des MBs MPEG-4 aux MBs H.264 utilisant l'architecture cascadée sur des séquences QCIF

Tiré de Lee (2004, p. 58)

H.264 (sortie) (%) / *MPEG-4 (entrée)*	**Mode inter**							**Mode intra**		**Skip**	**TOTAL**
				P8x8							
	16x16	16x8	8x16	8x8	8x4	4x8	4x4	16x16	4x4		
Inter16x16	**42.2%**	**12.4%**	**13.6%**	**9.6%**	1.7%	2.1%	1.2%	1.1%	0.8%	**15.5%**	100%
Inter8x8	**8%**	**11.8%**	**15.8%**	**28.6%**	**5.3%**	**6.7%**	**9.5%**	0.7%	13.5%	0.1%	100%
Skip	**5.8%**	1.1%	1.1%	0.4%	0.1%	0.1%	0%	0.5%	0%	**90.9%**	100%
Intra16x16	0%	0%	0%	2.6%	0%	0%	7.9%	**21.1%**	**68.4%**	0%	100%

Tableau 2.4 Pourcentage de conversion des MBs MPEG-4 aux MBs H.264 utilisant l'architecture cascadée sur des séquences CIF

Tiré de Lee (2004, p. 58)

H.264 (sortie) (%) / MPEG-4 (entrée)	Mode inter							Mode intra		Skip	TOTAL
				P8x8							
	16x16	16x8	8x16	8x8	8x4	4x8	4x4	16x16	4x4		
Inter16x16	**43.6%**	**13.5%**	**12.8%**	**14.4%**	4.3%	4.4%	2%	0.8%	0.5%	3.7%	100%
Inter8x8	**15.3%**	**13.5%**	**12.8%**	**27.5%**	**8.6%**	**8.7%**	**8.5%**	0.4%	4.7%	0.1%	100%
Skip	**7.2%**	0.9%	0.8%	0.2%	0%	0%	0%	1.5%	0%	**89.4%**	100%
Intra16x16	5.9%	3.5%	3.5%	8.2%	2.4%	1.2%	14.1%	**8.2%**	**52.9%**	0%	100%

La Figure 2.4 illustre la conversion des modes de MPEG-4 en entrée à H.264 en sortie (en se basant sur les statistiques des tableaux 2.3 et 2.4).

a. Si le MB en entrée est inter16x16, les modes inter16x16, inter16x8, inter8x16, inter8x8 et skip sont évalués.
b. Si le MB en entrée est inter8x8, les modes inter16x16, inter16x8, inter8x16, inter8x8 et ses sous modes sont évalués.
c. Si le MB en entrée est skip, les modes inter16x16 et skip sont évalués.
d. Si le MB en entrée est intra16x16, les modes intra16x16 et intra4x4 sont évalués.

En plus, étant donné que les modes Intra4x4 et Intra16x16 sont très coûteux en calcul du fait qu'ils calculent neuf modes de prédiction directionnelle et quatre modes de prédiction directionnelle respectivement, Lee et al. ont substitué la distorsion (RDO) par le calcul de l'erreur quadratique moyenne (EQM). Ainsi, les calculs du débit et de la distorsion sont évités. Finalement, pour atteindre la meilleure qualité possible, un raffinement de ±1 pixel est effectué sur les VMs.

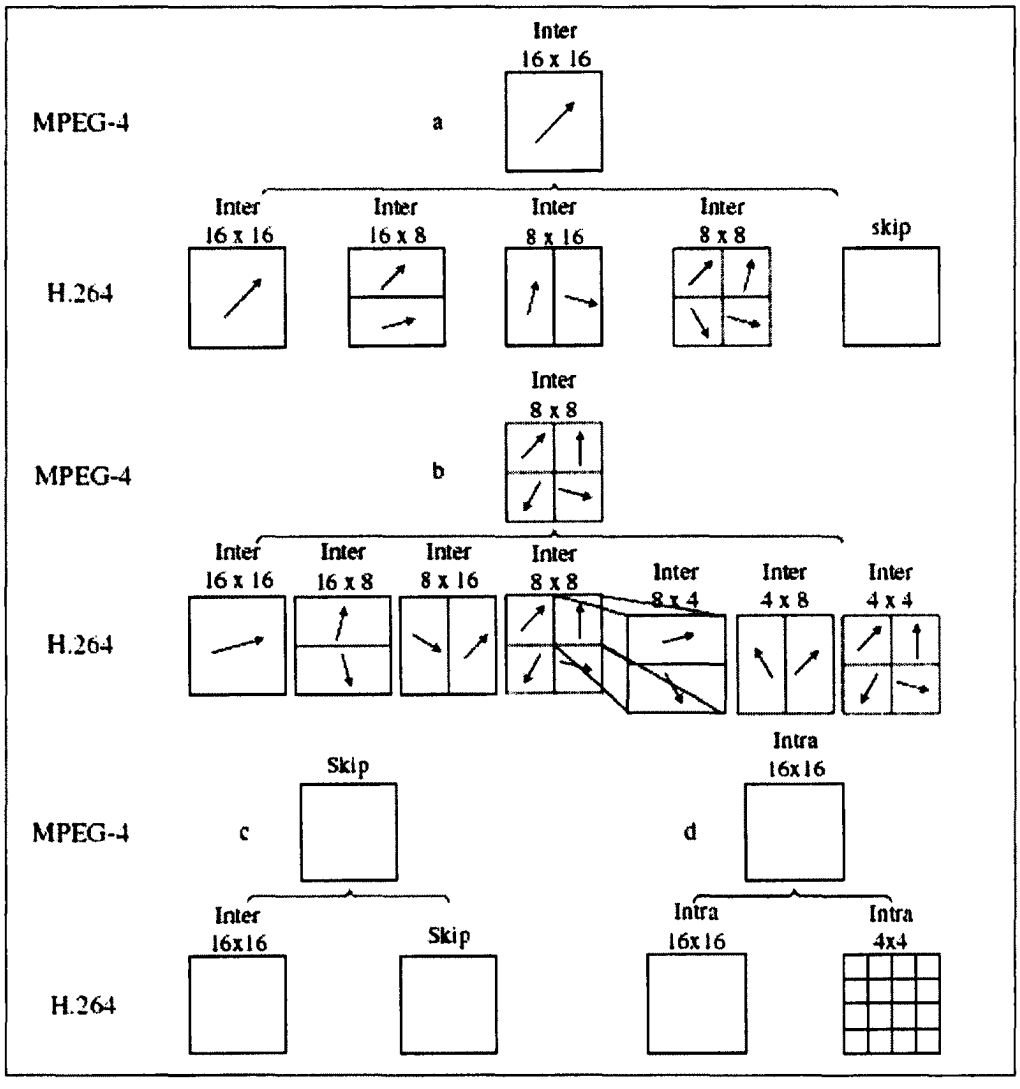

Figure 2.4 Modes de conversion H.264 à MPEG-4.
Tirée de Lee (2004, p. 59)

2.5.1.2 H.263 à H.264

L'idée de l'algorithme proposé par Lee [28] pour le changement de format de MPEG-4 vers H.264 peut être utilisée pour réaliser le changement de format de H.263 vers H.264.

Bialkowski a proposé dans [3] un transcodeur cascadé dans le domaine pixel (TCDP) optimisé pour le changement de format de H.263 à H.264. Le nouveau transcodeur utilise la somme absolue de l'énergie résiduelle Z en la comparant avec un seuil D choisi

empiriquement selon le paramètre de quantification (QP) pour transcoder les MBs intra. Les modes de prédiction sont aussi utilisés s'il s'agit de H.263 profil 3. Pour le transcodage des MBs inter, la méthode des statistiques est utilisée, mais en changeant le QP. Ensuite, un raffinement de ¼ pixel est effectué pour atteindre une qualité approximative à la vidéo en entrée.

2.5.1.3 MPEG-2 à H.264

Une extension de l'idée des statistiques et de la méthode proposée par Lee [28] pour le changement de format de MPEG-4 vers H.264 peut être effectuée pour réaliser le changement de format de MPEG-2 vers H.264.

Comme extension de sa nouvelle architecture de changement de format H.263 à MPEG-4, Bialkowski a proposé dans [3] le changement de format de MPEG-2 (progressif) vers H.264 par le biais du même principe utilisé pour H.263.

2.5.2 CF de H.264 vers MPEG-2, MPEG-4 et H.263

2.5.2.1 H.264 à MPEG-4

Hur et al. [13; 14; 15] ont introduit un algorithme basé sur les statistiques pour le changement de format de H.264 à MPEG-4. Le tableau 2.5 montre les résultats obtenus. Le mode inter8x8 a un pourcentage de 37.04% des MBs codés en H.264 et est transformé à skip, inter16x16 ou inter8x8. Le pourcentage des MBs skip est 24,86% et est mappé à skip ou inter16x16. Un prétraitement particulier est effectué sur les MBs skip de H.264 car les MBs skip PMV (*median Predicted Motion Vector*) ne sont pas supportés par MPEG-4 et seulement les MBs skip ayant le vecteur (0, 0) sont supportés. Inter16x16 occupe 18.77% et est transformé à skip ou inter16x16. Inter16x8 et inter8x16 occupent 6.81% et 6.46% respectivement et sont transformés à inter16x16 ou inter8x8. Intra16x16 et intra4x4 sont mappés à intra en MPEG-4. À la fin, un raffinement de ±1 pixel est effectué pour maintenir

la qualité originale. La Figure 2.5 détaille la liste des modes MPEG-4 probables pour chaque type de MB en H.264 lors du changement de format H.264 vers MPEG-4.

Tableau 2.5 Conversion des modes de macroblocks de H.264 vers MPEG-4
Tiré de Hur (2007, p. 2)

MPEG-4 / ***H.264***	**Intra**	**Skip**	**Inter16x16**	**Inter8x8**	**TOTAL**
Inter8x8 (fig. 2.4(a))	0.02%	**1.62%**	**22.52%**	**12.88%**	37.04%
Skip (fig. 2.4(b))	0%	**23.01%**	**1.73%**	0.12%	24.86%
Inter16x16 (fig. 2.4(c))	0%	**3.99%**	**13.36%**	1.42%	18.77%
Inter16x8 (fig. 2.4(d))	0.01%	0.96%	**4.61%**	**1.23%**	6.81%
Inter8x16 (fig. 2.4(e))	0.01%	0.76%	**4.30%**	**1.40%**	6.46%
Intra16x16 & Intra4x4 (fig. 2.4(f))	1.33%	0.32%	2.23%	2.17%	6.05%

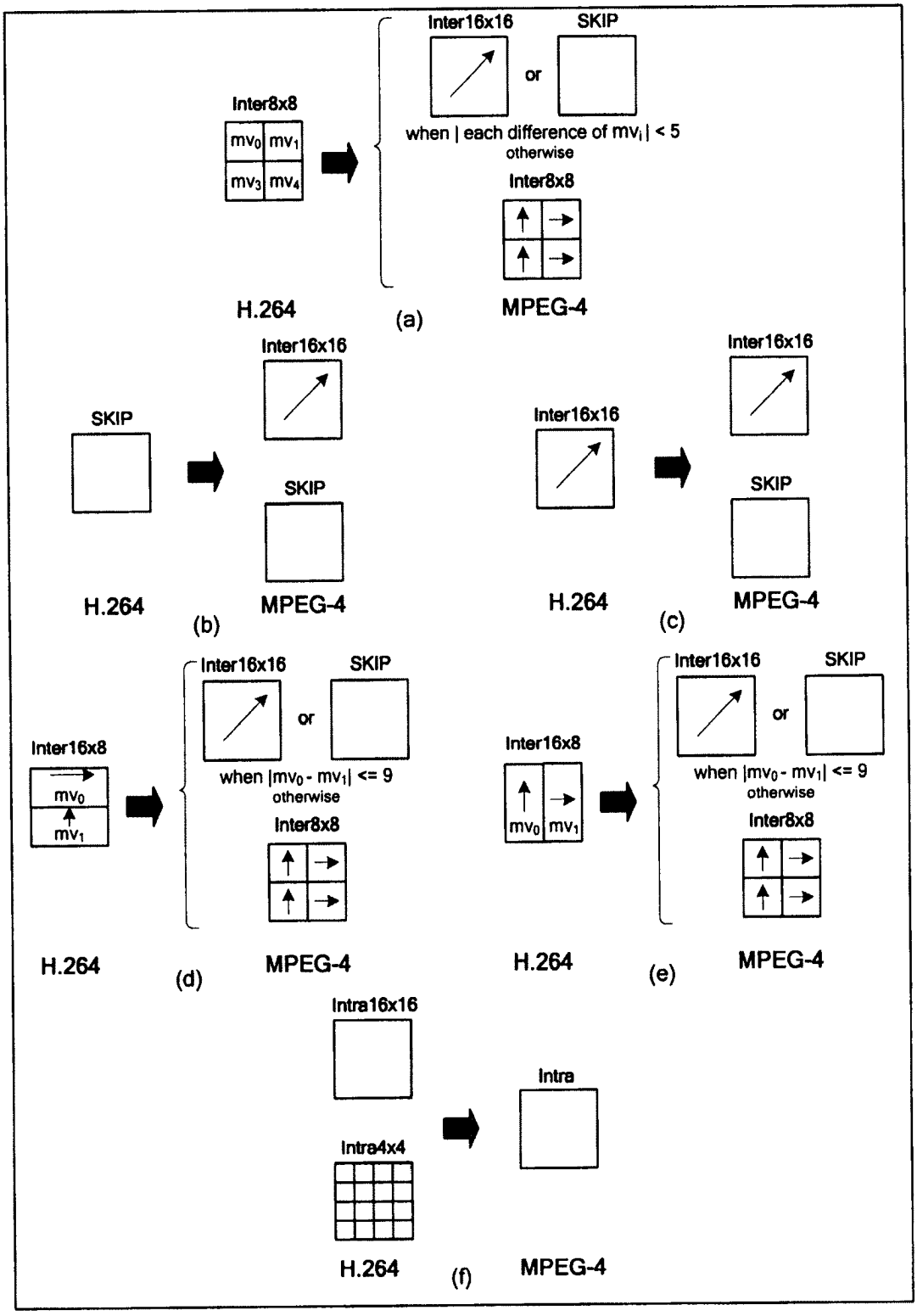

Figure 2.5 Modes de conversion de H.264 vers MPEG-4.

Adaptée de Hur (2007, p. 3)

2.5.2.2 H.264 à H.263

De même que pour MPEG-4, le CF de H.264 à H.263 peut être réalisé par la méthode proposée par Hur [15].

Bialkowski et al. [4; 5] ont proposé un TCDP capable de réaliser le changement de format de H.264 vers H.263. L'algorithme proposé réutilise les informations récupérées du décodeur H.264. Il compare le VM prédit avec le VM candidat de H.264. Ensuite, celui qui donne la plus basse erreur quadratique moyenne (EQM) est choisi et est raffiné selon une fenêtre de recherche adaptée. Un raffinement adapté de ± 3/2 ou 5/2 pixels est effectué.

2.5.2.3 H.264 à MPEG-2

L'algorithme proposé par Hur [15] et celui proposé par Bialkowski [4; 5] peuvent être étendus pour réaliser le changement de format de H.264 à MPEG-2.

2.5.3 CF entre H.263 et MPEG-4

Le point de départ du standard MPEG-4 était l'algorithme de H.263 baseline (profil 0). Ainsi, MPEG-4 est capable de décoder des trains de bits H.263 baseline. De ce fait, rares sont les travaux dans la littérature qui ont étudié le changement de format entre les deux standards. Dogan et al. [10] ont présenté un TCDP pour le CF entre H.263 et MPEG-4. Liang et al. [30] ont consacré plus de détail au CF de MPEG-4 à H.263. Ci-après l'algorithme utilisé par les deux travaux.

2.5.3.1 H.263 à MPEG-4

- Pour transcoder les MBs intra : les coefficients AC sont réencodés de nouveau en utilisant l'encodage à longueur variable (VLC) de MPEG-4.
- Pour transcoder les MBs inter : les VMs et les coefficients AC sont copiés tels quels.

- Pour transcoder les MBs skip : le MB est sujet à un décodage à longueur variable (VLD) puis un encodage à longueur variable (VLC).

2.5.3.2 MPEG-4 à H.263

- Pour transcoder les macroblocks (MBs) intra : les coefficients AC sont réencodés de nouveau en utilisant l'encodage à longueur variable (VLC) de H.263.
- Pour transcoder les MBs inter : les VMs et les coefficients AC sont copiés tels quels. S'il y a 4 VMs ou les VMs sont à l'extérieur de la surface limite, un raffinement des VMs est requis.
- Pour transcoder les MBs skip : le MB est sujet à un VLD puis un VLC.

2.5.4 CF entre MPEG-2 et H.263 (CF seulement de MPEG-2 à H.263)

Shanableh [42] a étudié un algorithme basé sur une architecture cascadée dans le domaine pixel (ACDP) pour changer pour la syntaxe du train de bits de MPEG-2 à H.263 profil 0 (voir Figure 2.6). Les trames B de MPEG-2 ont la caractéristique d'utilisation de deux VMs en utilisant deux références alors que H.263 profil 0 n'utilise pas les trames B. Dans ce qui suit, la notation $V_{X\rightarrow Y}^{fwd}$ est utilisée pour signifier un VM (MPEG-2) en avant de la trame X à la trame Y. De même, $V_{X\rightarrow Y}^{bwd}$ est utilisée pour signifier un VM (MPEG-2) en arrière de la trame X à la trame Y. Pour H.263 profil 0, qui utilise que les VMs en avant $V_{X\rightarrow Y}^{out}$ sera utilisée.

Le résumé de l'algorithme est comme suit :

- Pour la première trame B dans le groupe d'image (*group of pictures* GOP) du flux vidéo entrant, telle que la trame B_8, cette trame est convertie à une trame P en sortie, soit P_7, avec une prédiction de P_6. Le nouveau VM sortant $V_{6\rightarrow 7}^{out}$ peut être obtenu de l'un de ces VMs entrants :
  - À partir de son VM en avant, $V_{4\rightarrow 8}^{fwd}$
  - À partir de l'écart entre son VM en avant et son VM en arrière, $V_{4\rightarrow 7}^{fwd} + V_{7\rightarrow 8}^{bwd}$

- Pour la deuxième trame B dans le GOP du flux vidéo entrant, telle que la trame B_9, cette trame est convertie à une trame P en sortie, soit P_8, avec une prédiction de P_7. Le nouveau VM sortant $V_{7\rightarrow8}^{out}$ peut être obtenu du l'un de ces VMs entrants :
  - À partir de l'écart $V_{4\rightarrow9}^{fwd}$ - $V_{4\rightarrow8}^{fwd}$
  - À partir de $-(V_{7\rightarrow8}^{bwd} - V_{7\rightarrow9}^{bwd})$
  - À partir de $- V_{7\rightarrow9}^{bwd}$
- Pour la trame P dans le flux vidéo entrant, telle que la trame P_7, cette trame est convertie à une trame P en sortie, soit P_9. Le nouveau VM sortant $V_{8\rightarrow9}^{out}$ peut être obtenu du l'un de ces VMs entrants :
  - À partir de $- V_{7\rightarrow9}^{bwd}$
  - À partir de $V_{4\rightarrow7}^{fwd} - V_{4\rightarrow9}^{fwd}$
- Pour la trame intra dans le flux vidéo entrant telle que la trame I_{10}, cette trame est convertie à une trame P en sortie, soit P_{12}. Le nouveau VM sortant peut être obtenu du l'un de ces VMs entrants :
  - À partir de $- V_{10\rightarrow12}^{bwd}$

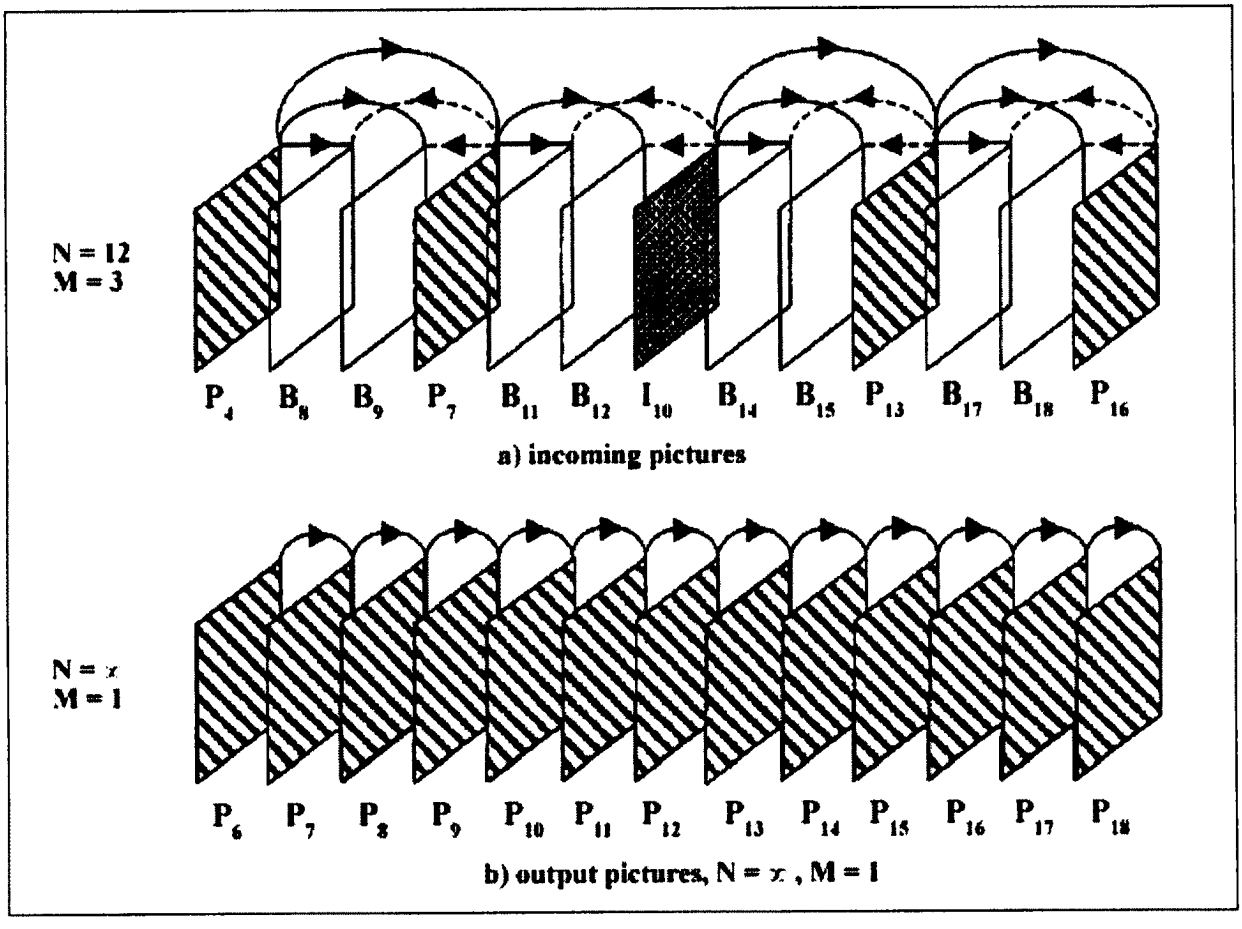

Figure 2.6 Trames de l'entrée MPEG-2 et de la sortie H.263.

Tirée de Shanaleh (2000, p. 102)

2.5.5 CF entre MPEG-2 et MPEG-4

La méthode présentée par Shanableh [42] pour changer la syntaxe du train de bits de MPEG-2 à H.263 peut être étendue pour le changement de format de MPEG-2 à MPEG-4.

Le transcodage de MPEG-2 entrelacé à MPEG-4 a été traité dans.[55]. Xin et al. ont élaboré un TCDP pour la réalisation du transcodeur.

2.6 Insertion de logo (IL)

Pour insérer un logo dans le domaine pixel, la méthode proposée par Panusopone [37], et illustrée par la Figure 1.19 (p. 43), est parmi les meilleures solutions. Cette solution introduit un effet minime sur la complexité de l'algorithme tout en gardant une meilleure qualité avec un débit binaire très proche du débit binaire de la vidéo compressée en entrée. La méthode de Panusopone réutilise les VMs et les modes de codage entrants pour les MBs non affectés par l'insertion. Pour la zone des MBs où le logo sera ajouté, c.-à-d. les MBs-logo, la méthode réduit la taille du logo au plus proche multiple de 16 (en largeur seulement) et positionne le logo dans le coin supérieur gauche du MB pour que l'influence de l'algorithme sur les MBs à modifier soit très petite. L'équation 2.5 est utilisée pour contrôler la translucidité du logo.

$$W_n = \alpha_n \cdot L_n + \beta_n \cdot I_n \tag{2.5}$$

L'impact de l'insertion sur la complexité de calcul est minimisé en utilisant soit le VM entrant soit le Vecteur (0,0) selon le paramètre alpha.

- Si $\alpha_n > 0.5$, c'est-à-dire que le MB-logo est dominé par le logo, le vecteur (0,0) sera choisi et aucun traitement d'estimation de mouvement (ME) n'est effectué. Dans ce cas, il faut revérifier le mode de codage du fait que les paramètres de codage ont changé.
- Si $\alpha_n < 0.5$, le VM entrant est choisi.

En plus des MBs-logo, les MBs qui avaient des VMs qui pointaient vers les MBs-logo sont aussi touchés par l'insertion du logo. Pour ces MBs, les VMs peuvent être incorrects après l'insertion et doivent être recalculés. Pour résoudre ce problème, chaque MB qui ne pointait pas vers les MBs-logo va recevoir son VM entrant, et les autres MBs qui pointaient vers les MBs vont avoir des vecteurs (0,0).

L'insertion du logo induit généralement à une augmentation du flux de données de la vidéo transcodée. Ceci est résolu en jouant sur le paramètre de quantification (QP).

2.7 Conclusion

Dans ce chapitre, nous avons analysé les détails de chaque opération de transcodage vidéo incluant la réduction du débit binaire (RDB), la réduction de la résolution spatiale (RRS), la réduction de la résolution temporelle (RRT), le changement de format (CF) et l'insertion de logo (IL). Les algorithmes présentés sont flexibles du fait qu'ils réalisent le transcodage dans le domaine pixel. En plus, en réutilisant certaines informations récupérées du flux entrant, la complexité de ces transcodeurs peut significativement baisser. Cependant, ces transcodeurs se sont concentrés sur la réalisation d'une seule opération, ce qui force à en créer autant de transcodeurs pour chaque cas d'utilisation. Ceci augmente considérablement l'effort de développement du logiciel et rend sa maintenance très complexe.

CHAPITRE 3

ANALYSE DES TRANSCODEURS VIDÉO RÉALISANT PLUS D'UNE OPÉRATION

3.1 Introduction

Le marché du multimédia est très compétitif. Pour avoir une place dans ce marché, les compagnies se doivent d'être très flexibles en supportant différentes normes, en proposant différentes résolutions, etc.

Dans le transcodage vidéo, il arrive souvent qu'on ait besoin de demander au transcodeur d'exécuter plusieurs opérations de transcodage vidéo simultanément. Cependant, la plupart des travaux réalisés dans le domaine du transcodage vidéo traitent chaque fonctionnalité isolément. Dans ce qui suit, nous allons nous concentrer sur tous les travaux qui ont été réalisés dans la littérature et qui ont abordé le sujet du transcodage hétérogène -dans les deux domaines spatial ou transformé-, c.-à-d. la réalisation de plusieurs opérations de transcodage vidéo et spécifiquement la réduction du débit binaire, la réduction de la résolution spatiale, la réduction du nombre de trames par seconde, le changement de la syntaxe du flux vidéo et l'insertion de logo. Il est à noter que les gains en vitesse sont calculés de la manière suivante. Si l'algorithme proposé réalise le transcodage en 1 seconde et l'algorithme avec lequel on compare le réalise en 3 secondes, le gain est 3/1 = 3 fois. Donc l'algorithme proposé est trois fois plus rapide –c.-à-d. les gains sont 300%- ou ~67 % moins complexe que la référence.

3.2 Le transcodeur de Shanableh

Le premier transcodeur que nous présentons ici est appelé transcodeur de Shanableh. Shanableh et al. [42] ont implémenté un transcodeur basé sur une architecture cascadée dans le domaine pixel (ACDP) visant la réduction du débit binaire en changeant le format vidéo de MPEG-1/2 à H.261/H.263 avec une réduction de la résolution spatio-temporelle. Cette architecture est illustrée par la Figure 3.1.

Le transcodeur effectue un décodage dans le domaine pixel du flux vidéo entrant. Les trames décodées sont sujettes à une réduction de la résolution spatiale (RRS) ou une réduction de la résolution temporelle (RRT) puis à un changement de format (CF) de MPEG-1/2 à H.261/H263, en utilisant les paramètres extraits du train de bits en entrée (*bitstream*). Ces paramètres englobent entre autres la taille du GOP, la résolution, les VMs et les modes de MBs. Après la réalisation de ces opérations, l'architecture raffine les vecteurs de mouvements (VMs) pour augmenter la qualité de la vidéo sortante. À la fin, les trames avec la syntaxe de H.263 de résolution spatiale ou temporelle réduite sont réencodées sans utiliser une nouvelle estimation de mouvement (ME), ce qui permet d'avoir une réduction des calculs d'au moins 70% par rapport à l'architecture cascadée de référence.

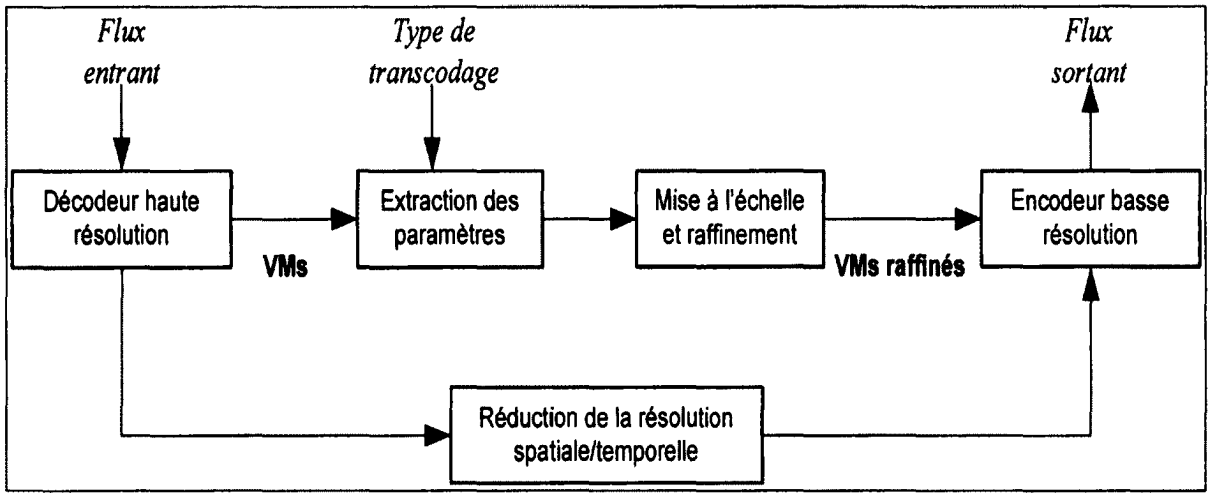

Figure 3.1 Architecture de Shanableh.

Adaptée de Shanableh (2000, p. 102)

Pour changer le format de MPEG-2 à H.263, nous reprenons ici l'idée principale décrite dans la section 2.4.4. Shanableh a proposé plus d'une solution pour chaque changement de type trame, c.-à-d. d'une trame B à une trame P, d'une trame P à une trame P et d'une trame I à une trame P.

Pour réaliser la RRS, Shanableh a présenté une préférence pour la méthode "*majority voting*" qui, selon l'auteur, aboutit au même résultat de la méthode médiane. Cette dernière donne un

VM issu de la somme des VMs d'un (MB) ayant des directions similaires en exploitant la corrélation entre eux. La méthode "*majority voting*" est définie par l'équation 3.1 :

$$V = \frac{1}{m}\sum_{i=1}^{m} V_i, \quad m \leq 4 \tag{3.1}$$

où V représente le VM final, V_i représente les VMs ayant la même direction et m est le nombre maximal des VMs V_i. Si m égale 4, la méthode "*majority voting*" devient une simple moyenne.

La réduction du nombre de trames est implémentée par la méthode *Telescopic Vector Composition* (TVC), qui donne selon l'auteur un résultat très proche de la méthode *Forward Dominant Vector Selection* (FDVS).

Le transcodeur de Shanableh aboutit à une réduction très importante de la complexité de calcul étant donné que l'estimation de mouvement (ME) qui nécessite généralement 70% du temps d'encodage est évitée. Cependant, cet algorithme a traité les trois cas un par un sans donner une recette détaillée de la manière de les combiner ensemble, et ce quoique les simulations comparent l'effet de l'hétérogénéité des opérations par rapport à l'effet de traiter chaque cas isolément.

3.3 Le transcodeur de Feamster

Feamster et al. ont développé un transcodeur cascadé dans le domaine pixel (TCDP) pour le transcodage MPEG-2 entrelacé à H.263 avec une réduction spatio-temporelle [11] et [51]. La Figure 3.2 montre le transcodeur de Feamster.

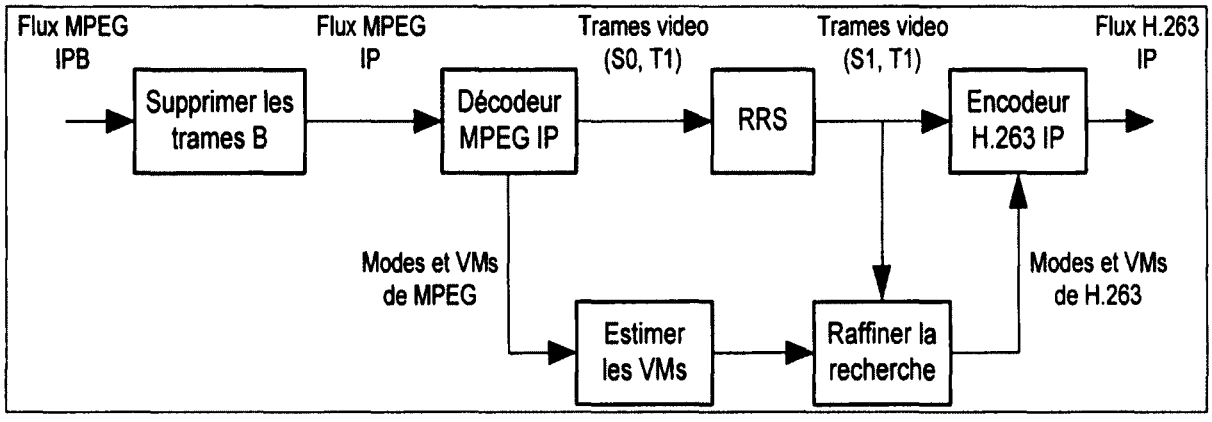

Figure 3.2 Architecture de Feamster.

Adaptée de Feamster (1999, p. 168)

L'objectif des travaux de Feamster et al. est d'établir un transcodeur efficace pour convertir la syntaxe de la vidéo entre MPEG-2 et H.263 tout en maintenant une haute qualité et une faible complexité du transcodeur. Ce transcodeur diffère de celui présenté par Shanableh dans le fait qu'il supporte la syntaxe entrelacée de la norme MPEG-2. L'ordre des opérations dans ce transcodeur est comme suit :

1. Réduction de la résolution temporelle (RRT) : les trames B sont supprimées avant de décoder le train de bits en entrée car elles ne contiennent aucune information de mouvement. Le résultat de la RRT est un flux vidéo MPEG avec des trames I ou P seulement. À ce moment, les trames I et P sont décodées dans le domaine pixel.
2. Réduction de la résolution spatiale (RRS) : la résolution spatiale du flux vidéo est mise à l'échelle par un facteur de deux. La syntaxe de MPEG-2 entrelacé permet de faire, d'une façon intuitive, une réduction spatiale en ne gardant que les images de haut "*top field*" mais en réduisant leur échelle par deux. Ainsi, le transcodeur permet passer de CCIR (720x480) à 352x240. Pour les lignes horizontales, on échantillonne les lignes alternatives puis on élimine les 8 lignes à droite. Pour les lignes verticales, on passe de 480 à 240 pixels en ne gardant que les pixels de l'image de haut (*top field*).
3. Changement de format (CF) : la syntaxe de MPEG-2 entrelacé est transformée en H.263 progressif. Deux parcours sont possibles, soit on change les images I à des trames P, soit on change les images P à des trames P.

4. Raffinement : finalement les vecteurs de mouvements (VMs) issus de ces trois opérations sont raffinés par une fenêtre de ½ pixel.

Le transcodeur de Feamster mène à une qualité très proche de l'architecture cascadée de référence avec une perte de 0.5 dB en PSNR, mais elle est 5 fois plus vite que cette dernière en utilisant le code de référence de H.263. La limite de ce transcodeur est la supposition que l'entrée et la sortie sont toujours MPEG-2 et H.263, respectivement. Par exemple, si on veut étendre cette architecture au transcodage de MPEG-2 et MPEG-4 profil simple visuel avec une mise à l'échelle spatiale et temporelle, la méthode de suppression directe des trames B n'est pas valide. De ce fait, du point de vue de la flexibilité, sur lequel nous avons choisi les architectures dans le chapitre 2, le transcodeur de Shanableh où le CF est effectué en utilisant une technique de changement de format au lieu de supprimer les trames bidirectionnelles (trames B) donne plus de flexibilités pour traiter la conversion de format entre d'autres standards ou même pour mener le transcodeur à traiter plus de cas possibles, comme l'insertion de logo.

3.4 Le transcodeur de Vetro

Le but des travaux de Vetro et al. [50] est de proposer une nouvelle technique de transcodage efficace pour la réduction de la résolution spatio-temporelle. Cette nouvelle technique peut être appliquée pour le même format ou entre différents formats (MPEG-2 vers MPEG-4 profil simple). Contrairement aux deux transcodeurs présentés (Shanableh et Feamster), le transcodeur proposé par Vetro et al. est basé sur une architecture ouverte avec une technique de compensation de la dérive. Cette architecture, illustrée par la Figure 3.3, utilise une seule compensation de mouvement et une seule mémoire-tampon par rapport aux deux architectures précédentes qui en utilisent deux. Le flux vidéo entrant est décodé et les trames sont stockées dans la mémoire-tampon après une MC. Ensuite, après la réalisation de la réduction spatio-temporelle avec CF, la résiduelle est ré-calculée en se basant sur la somme des résiduelles de la trame courantes plus les trames sautées. Ce mappage est effectué par le bloc "Mappage de la résiduelle". Le bloc "Conversion Inter-à-Intra" permet de contrôler le

changement des MBs à des MBs intra en synchronisant avec le bloc "Contrôle du débit" en se basant sur le facteur β qui représente le pourcentage des MBs intra dans la trame. Après la réduction spatio-temporelle et le mappage de la résiduelle, le flux vidéo sortant est quantifié et encodé avec VLC.

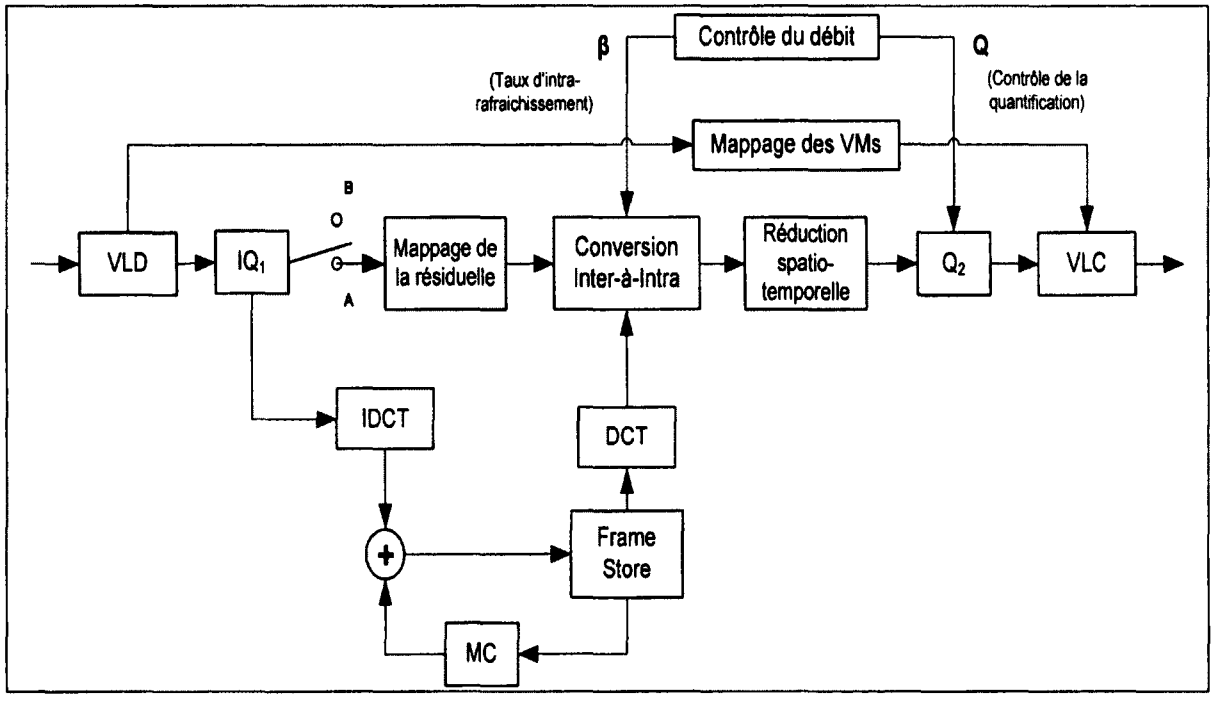

Figure 3.3 Architecture d'intra rafraîchissement.

Adaptée de Vetro (2002, p. 724)

La mise à l'échelle spatiale ou temporelle est effectuée dans ce transcodeur par un facteur 2. Une réduction de la résolution temporelle (RRT) est effectuée en premier lieu et les vecteurs de mouvements (VMs) sont estimés à partir de la trame conservée. Ce traitement est réalisé par la méthode "*majority voting*". Ensuite, les VMs issus de cette opération sont réestimés pour la nouvelle résolution spatiale réduite. La méthode de la médiane est utilisée dans ce cas avec une division par 2 des VMs. Si le flux vidéo MPEG-2 en entrée est entrelacé, la RRS est implémentée en ne gardant que l'image de haut "*top field*" et en utilisant ses VMs avec une mise à l'échelle par 2. À la fin, et vu que cette architecture emploie une seule mémoire (voir tableau 3.1 et Figure 3.3), une réestimation de la résiduelle est faite dans le domaine DCT.

Tableau 3.1 Comparaison de l'architecture d'intra-rafraîchissement avec l'ACDP
Tiré de Vetro (2002, IV-726)

Architecture de transcodage	**DCT/IDCT**	**Boucle de compensation/estimation de mouvement (MC/ME)**	**Mémoire (Frame store)**
Référence (ACDP)	4	2	2
Intra-rafraîchissement	2	1	1

Dans l'architecture d'intra rafraîchissement, les VMs sont réutilisés, la résiduelle est recalculée et aucune estimation de mouvement (ME) n'est faite côté encodeur. En plus, comparée à l'architecture cascadée modifiée telle que celle de Shanableh ou Feamster, l'architecture de Vetro utilise moins de DCT/IDCT et de mémoire. Cependant, le gain obtenu par cette architecture est ordinaire (1.5) et le débit binaire est plus élevé à cause du transfert d'un certain pourcentage de MBs inter en intra. Aussi, il y a une perte de qualité significative en raison du mappage de la résiduelle et de l'absence du raffinement des VMs.

3.5 Le transcodeur de Xin

Les travaux de Xin et al. [55] présentent une architecture cascadée dans le domaine pixel (ACDP) pour le transcodage entre MPEG-2 entrelacé et MPEG-4 profil simple avec une réduction spatio-temporelle. La Figure 3.4 montre cette architecture.

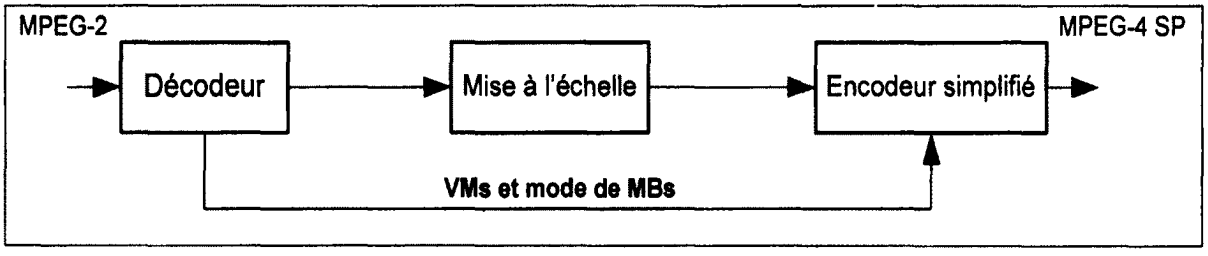

Figure 3.4 Architecture de Xin.
Tirée de Xin (2002, p. 7)

L'ordre des opérations dans l'architecture de Xin est le suivant :

1. Le flux vidéo en entrée est complètement dans le domaine pixel et les méta-informations nécessaires sont stockées (VMs, modes, résolution temporelle, résolution spatiale, GOP, etc.).
2. Le transcodeur effectue une réduction temporelle par un facteur de 2, et les nouveaux VMs des trames non sautées sont stockés. Le tableau 3.2 illustre un groupe d'images (group of pictures GOP) en format MPEG-2 entrelacé et son équivalent en sortie MPEG-4. Le GOP en entrée est constitué de 16 images.
3. Une réduction spatiale est réalisée permettant de passer de CCIR (720x480) à CIF (352x288). Le transcodeur supprime la 1ère image (*field*) de la trame en entrée. Les VMs des trames non sautées sont réutilisés pour un mappage des VMs dans la taille réduite en utilisant la médiane puis une réduction par 2.
4. La conversion de format est faite selon trois partons (tableau 3.2), deux pour transformer une image B vers une trame P et un patron pour transformer une image P vers une trame P. À la fin, le transcodeur raffine les VMs par le biais d'une petite fenêtre de l'ordre de 0.5 pixel.

Tableau 3.2 Conversion des types de trames MPEG-2 vers les types de trames MPEG-4
Tiré de Xin (2002, p. 6)

Numéro en entrée	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Entrée	I	B	B	P	B	B	P	B	B	P	B	B	P	B	B	I	P
Sortie	I		P		P		P		P		P		P		P		P
Numéro en sortie	0		1		2		3		4		5		6		7		8
Patron de conversion			1		2		3		1		2		3		1		2

Le transcodeur de Xin mène à un compromis qualité/complexité où la qualité obtenue est supérieure à la qualité obtenue par l'architecture cascadée de référence –pour certaines séquences-, et le gain en vitesse de cette architecture est très important (5 fois plus rapide ou 80% moins de calculs par rapport à l'architecture cascadée avec ré-estimation des VMs).

3.6 Comparaison entre les différents travaux

Dans cette section, une comparaison a été réalisée entre les différentes architectures présentées précédemment, c.-à-d. celles de Shanableh, de Feamster, de Vetro et de Xin. Le tableau 3.3 récapitule cette comparaison.

3.7 Conclusion

Dans ce chapitre, nous avons présenté quatre travaux qui ont réalisé le transcodage hétérogène entre différentes fonctionnalités : la réduction du débit binaire, le changement de format, la réduction spatiale ou temporelle et l'insertion de logo. Ce qui est à remarquer de ces quatre travaux est qu'aucun d'entre eux n'a considéré le problème du transcodage entre toutes les normes existantes : H.263, H.264, MPEG-2 et MPEG-4 ; et la fonctionnalité de l'insertion de logo. Ces deux problèmes nécessitent d'être pris en considération dans les transcodeurs hétérogènes surtout avec les innovations majeures du nouveau standard H.264 qui utilise une prédiction de ¼ de pixel et une estimation de mouvement/compensation de mouvement (ME/MC) de taille variable, d'un bloc de 16x16 jusqu'à seize blocs de 4x4.

Tableau 3.3 Tableau comparatif des différents travaux

	Vetro et al.	**Shanableh et al.**	**Feamster et al.**	**Xin et al.**
Objectif	Réduction spatio-tempo-relle pour le même format ou entre deux formats différents (MPEG-1/2 à MPEG-4)	CF de MPEG-1/2 à H.261/H.263 avec réduction spatio-temporelle.	CF MPEG-2 entrelacé à H.263 avec une RDB (réduction de la résolution spatio-temporelle)	CF de MPEG-2 entrelacé à MPEG-4 profil simple avec réduction de la résolution spatio-temporelle.
Facteurs de RRT	Entier (2, 3, etc.).	Entier.	3.	2.
Facteur de RRS	2	2	2	2
Famille d'architecture	Architecture ouverte	ACDP.	ACDP	ACDP
Ordre des opérations	1. RRT 2. RRS 3. CF	Non mentionné.	1. RRT 2. RRS 3. CF	1. RRT 2. RRS 3. CF
Méthode de CF	Non mentionnée.	Selon le patron illustré par la Figure 2.4.	Convertir certaines trames I et P de MPEG-2 à des trames P de H.263.	La méthode de CF est illustrée par le tableau 3.1.
Méthode de RRS	Mappage des VMs de 4:1 et 1:1 en se basant sur la variance des 4 VMs. Dans le cas de MPEG-2 entrelacé, le dés-entrelacement est implémenté en gardant les VMs de l'image de haut (*top field*). Dans les deux cas, les VMs sont divisés par 2.	Mappage des VMs 4:1 et 1:1 en utilisant la médiane puis un sous-échantillonnage par 2.	L'architecture permet de passer de CCIR (720x480) à 352x240 en gardant l'image de haut seulement. Les VMs de l'image de haut sont réutilisés après une division par 2.	Le transcodeur supprime la 1ère image (*field*) de la trame en entrée puis reformule les VMs à partir de l'image sauvegardée en utilisant la médiane puis un sous-échantillonnage par 2.
Méthode de RRT	"*majority voting*"	"*Telescopic vector composition*".	Suppression des trames B	Voir tableau 3.1.
Raffinement des VMs	Non utilisé	± 0.5 pixel	± 0.5 pixel	± 0.5 pixel

CHAPITRE 4

SYSTÈME UNIFIÉ DE TRANSCODAGE VIDÉO

4.1 Introduction

Dans ce chapitre, nous formulons le problème des transcodeurs vidéo. Ensuite, nous présentons notre système de transcodage vidéo pour pallier les différentes limitations des transcodeurs vidéos. À la fin, nous montrons les nouveaux algorithmes développés pour les deux types de scénarios possibles : une seule opération de transcodage vidéo ou un ensemble d'opérations.

4.2 Formulation du problème

Un transcodeur vidéo transforme une vidéo d'un format en entrée à un autre format en sortie. Le format de la vidéo est régi par des caractéristiques telles que : le débit binaire, la résolution spatio-temporelle et le format d'encodage.

Dans les transcodeurs vidéo présentés dans la littérature, on trouve généralement un transcodeur vidéo dédié à chaque cas d'utilisation. Ainsi, la plupart des transcodeurs proposés ne peuvent supporter qu'une une seule fonctionnalité, c.-à-d. un transcodeur pour la réduction de la résolution spatiale [6; 42; 56], un transcodeur pour le changement entre deux standards de compression [3; 10; 11; 13], un transcodeur pour la réduction du débit binaire [6; 36; 57], etc. Rares sont les transcodeurs qui ont essayé de voir le problème du transcodage vidéo d'une manière générique, c'est-à-dire de proposer une solution capable d'effectuer n'importe quelle combinaison de ces opérations (nous avons présenté ces rares transcodeurs au chapitre précédent). De plus, en ce qui concerne le changement de format, c.-à-d. entre MPEG-2, MPEG-4, H.263 et le nouveau standard MPEG-4 Part 10 (H.264/AVC), un système générique qui est capable de réaliser une transformation entre tous ces formats est introuvable. La réalisation d'un tel système générique qui permet de tout transcoder est très complexe, car les caractéristiques du format en sortie sont variables et inconnues.

Tels que nous avons vu dans le chapitre précédent, quelques travaux seulement ont traité le sujet du transcodage hétérogène en focalisant sur les opérations suivantes : la réduction

spatio-temporelle et le changement de format entre deux standards bien définis. Shanableh [42] et Feamster [11] ont traité le changement de format de MPEG-2 à H.263, tandis que Xin [55] et Vetro [50] ont traité MPEG-2 à MPEG-4. Quoique ces travaux aient essayé de proposer des méthodes de transcodage vidéo aptes à exécuter plus d'une opération dans le même transcodeur vidéo, certaines opérations et certains formats y manquaient. Le nouveau standard H.264/AVC n'a pas été pris en considération dans ces travaux du fait qu'il était encore en cours de standardisation. Ce format avec toutes les nouvelles caractéristiques qu'il apporte, représente un grand défi pour la réalisation d'une architecture générique.

Dans la section suivante, nous présentons notre solution à tous ces problèmes en mettant en place un nouveau système de transcodage vidéo.

4.3 Système unifié

Le nouveau système de transcodage vidéo, appelée système unifié, est basé sur une architecture cascadée dans le domaine pixel (ACDP), flexible et capable de réaliser une ou plusieurs opérations de transcodage vidéo selon les paramètres de sortie désirés. Les fonctionnalités de transcodage supportées par ce système sont :

- Adaptation de la résolution spatiale.
- Adaptation de la résolution temporelle.
- Adaptation du débit binaire.
- Insertion de logo.
- Adaptation du format.
- Amélioration de la qualité (ou *visual enhancement*).
- Désentrelacement.
- Résistance aux erreurs.

Notre système unifié de transcodage vidéo permet de réaliser une seule opération à la fois, ou une combinaison d'opérations telle que l'adaptation de la résolution spatiale avec une insertion de logo, l'adaptation du format avec une adaptation du débit binaire, l'adaptation du format avec une adaptation de la résolution spatiale et une adaptation du taux de trames de la vidéo, etc. Ces combinaisons sont réalisées selon les paramètres de la séquence vidéo désirés

en sortie, c.-à-d. transformer n'importe quel train de bit en entrée ayant un format F1 à n'importe quel format en sortie ayant un format F2. Les formats F1 et F2 ont probablement différents débits binaires, différentes résolutions spatiales ou temporelles, différentes normes de compression, avec ou sans logo inséré. En plus, nous traitons dans notre système unifié le transcodage entre les principaux standards utilisés par les différentes applications multimédias, que ce soit MPEG-2, MPEG-4, H.263 ou H.264.

Notre transcodeur se veut :

- Un système unifié capable de réaliser n'importe quelle combinaison d'opérations de transcodage vidéo.
- Un système très optimisé en calcul et qui donne une vidéo avec une très haute qualité en exploitant au maximum les informations récupérées de la vidéo décompressée.
- Un système extensible, capable de rencontrer les exigences futures telles qu'une extension d'une norme de compression, ou même le développement d'une nouvelle norme probablement avec un ensemble de nouveautés et caractéristiques. Ainsi, au cas où un nouveau cas d'utilisation vient s'ajouter à la liste des cas, le développement et l'intégration du nouveau cas auront un impact mineur sur la conception du système lui-même, sur la taille du code à ajouter et sur la maintenabilité du système en entier.

La notion de généricité, étant donné qu'elle est aujourd'hui très émergente et très utilisée notamment en génie informatique, en génie logiciel et en technologies de l'information en raison de son impact majeur sur le développement du logiciel dans toutes ses phases, de l'analyse des besoins jusqu'à la réalisation et les tests, et par la suite la maintenance du logiciel, est employée dans notre transcodeur unifié pour la résolution de notre problème.

La Figure 4.1 illustre la nouvelle architecture unifiée de transcodage vidéo. Dans ce qui suit, nous présentons les différents blocs de notre architecture unifiée. Ensuite, nous décrivons les divers blocs qui interviennent suite à une opération ou un ensemble d'opérations désirées.

4.3.1 Description des blocs

4.3.1.1 Bloc de décodage/codage entropique

Ces deux blocs représentent le premier et le dernier bloc dans le processus de transcodage vidéo. Le bloc du décodage entropique décode le flux vidéo entrant en employant l'inverse du codage entropique pour obtenir un flux vidéo non compressé. Le bloc de codage entropique utilise des algorithmes de compression sans perte tels que *Variable length coding* (VLC), *Context-adaptive variable-length coding* (CAVLC), *Context-adaptive binary arithmetic coding* (CABAC), etc., pour réduire la taille du train de bits (*bitstream*) envoyé.

4.3.1.2 Bloc de quantification/quantification inverse

La quantification est appliquée sur l'ensemble des coefficients DCT et permet de réduire la taille du fichier. Cela entraîne toutefois une dégradation de qualité plus ou moins importante selon la valeur du paramètre de quantification choisi. Il est à noter que chaque format utilise une méthode de quantification différente.

4.3.1.3 Bloc de transformée/transformée inverse

Ce bloc permet de décorréler les coefficients (pixels) du même bloc pour pouvoir effectuer une quantification sur la vidéo. La DCT est la transformée la plus utilisée dans le domaine de codage et de transcodage vidéo. Souvent utilisée sur des blocs de 8x8 pixels dans les standards H.263, MPEG-2 et MPEG-4, le plus récent standard H.264 emploie une variante améliorée qui agit sur des blocs 4x4.

4.3.1.4 Bloc de compensation de mouvement

Ce bloc est employé deux fois dans le processus de transcodage. Du côté du décodeur, il est utilisé pour reconstruire la trame-inter courante à partir d'une trame ou d'un ensemble de trames qui sont stockées dans la mémoire tampon. Du côté de l'encodeur, le bloc de compensation de mouvement est nécessaire pour synchroniser l'encodage de la vidéo par le

transcodeur et le décodage de la vidéo par le décodeur du terminal final. Les deux blocs de compensation de mouvement peuvent avoir un comportement légèrement différent vis-à-vis du standard de compression utilisé en entrée et en sortie.

4.3.1.5 Bloc de mémoire-tampon

Les trames-intra ou inter récupérées par le décodeur sont stockées dans la mémoire-tampon (*Frame Store*) pour servir comme une référence pour le décodage des trames futures. Du côté de l'encodeur, les trames-intra ou inter même stockées dans la mémoire-tampon sont aussi utilisées pour l'encodage des trames futures. Ce bloc est lié directement aux spécifications du standard de compression. Par exemple, pour H.264 la taille maximale de la mémoire-tampon est 16 trames (ou 32 images en entrelacé).

4.3.1.6 Bloc de passage des paramètres

On y stocke toutes les métadonnées nécessaires pour accélérer le mécanisme de transcodage vidéo, tout en maintenant une très haute qualité. Parmi les métadonnées :

- Le type de la trame (trame I, P, B, etc.).
- Le type des plans (YUV, RGB, etc.).
- Le nombre de macroblocks (MBs).
- Les vecteurs de mouvements (VMs).
- Les modes de MBs.
- Le paramètre de quantification (QP).

4.3.1.7 Bloc des opérations

Le bloc des opérations est le bloc le plus important dans la nouvelle architecture de transcodage vidéo. Il réalise l'opération ou l'ensemble des opérations exigées pour rencontrer les capacités du système de l'usager final. Ainsi, ce bloc convertit la vidéo compressée d'un format en entrée au format désiré en sortie.

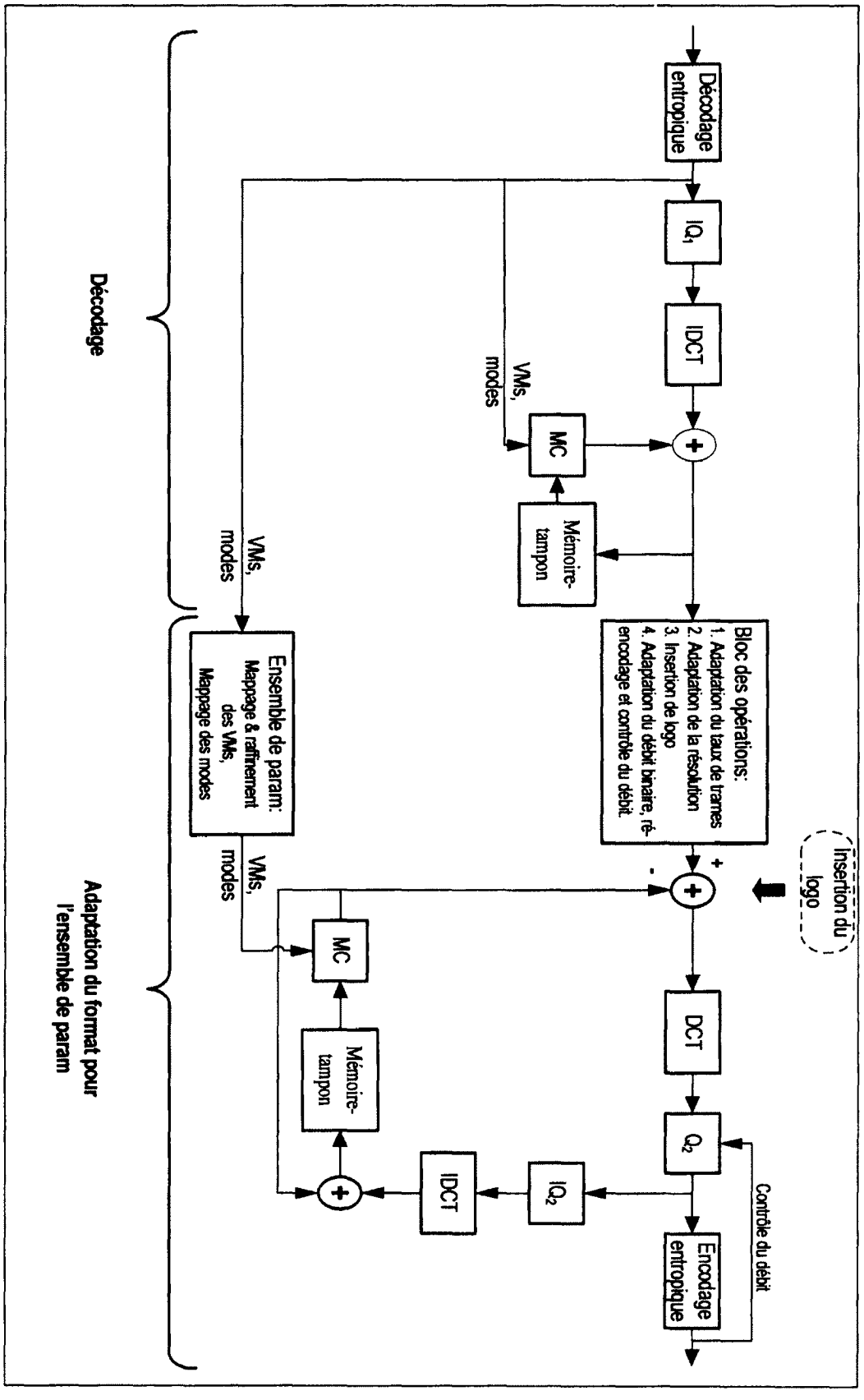

Figure 4.1 Architecture de transcodage vidéo unifiée.

4.3.2 Blocs invoqués pour la réalisation d'une seule opération

Dans notre système unifié, les différentes fonctionnalités du transcodage vidéo requièrent un décodage complet dans le domaine pixel de la vidéo compressée en entrée pour récupérer toutes les trames et les informations des VMs et des modes de MBs. Le transcodeur implémente l'opération souhaitée puis réencode la vidéo. Notons que le décodage est typiquement six fois moins complexe en calculs que l'encodage et qu'il importe de réduire la complexité de ce dernier.

L'opération de décodage dans le domaine pixel est commune entre toutes les fonctionnalités. Ainsi, les trames en entrée sont sujettes à un décodage entropique (VLD/CABAC/CAVLC), puis une déquantification (IQ_1) et une transformée en cosinus discrète inverse (IDCT). Des trames intra ou des trames résiduelles sont obtenues après la IDCT. De ce fait, une mémoire-tampon et une compensation de mouvement sont nécessaires pour la reconstitution des trames inter à partir de la résiduelle et des méta-informations. Par la suite, toute opération peut être effectuée après la réalisation du décodage.

Après avoir implémenté l'opération demandée, les trames, les modes de MBs et les VMs adaptés sont réencodés. L'opération de réencodage est presque similaire entre toutes les fonctionnalités. L'encodeur effectue de nouveau une transformée en cosinus discrète (DCT), une quantification (Q_2) et un encodage entropique (VLC/CABAC/CAVLC). Dans certaines opérations, les modes des VMs peuvent subir une réestimation et les VMs peuvent nécessiter un raffinement. L'encodeur contrôle aussi le débit binaire du flux vidéo sortant et l'adapte selon les besoins ou selon les limites du réseau de transport.

4.3.2.1 Blocs invoqués pour l'adaptation du taux de trames

L'adaptation du taux de trames du flux vidéo compressé est réalisée après le décodage dans le domaine pixel. Le bloc responsable de la réalisation de cette fonctionnalité est le bloc adaptation du taux de trames. Ce dernier fait appel au bloc de mappage des vecteurs de mouvements pour effectuer le mappage.

Les vecteurs de mouvements (VMs) sont mappés de la trame courante à la trame précédente non sautée en utilisant les VMs de la trame sautée (des trames sautées). Cette opération nécessite la sauvegarde temporaire des VMs des trames sautées. Dans le cas contraire, c.-à-d. l'augmentation du taux de trames, les VMs des trames introduites sont obtenus par une interpolation dans la direction du mouvement.
Par la suite de l'opération d'adaptation du taux de trames, les VMs sont raffinés par une fenêtre de recherche de taille très petite et les modes sont réestimés par l'encodeur.

4.3.2.2 Blocs invoqués pour l'adaptation de la résolution spatiale

Après le décodage dans le domaine des pixels et la récupération des informations extraites du train de bits en entrée, l'adaptation de la résolution spatiale est implémentée par le bloc portant le même nom. Ce bloc fait appel au bloc de mappage des VMs pour transformer les VMs de la taille originale à des VMs de la taille désirée. Puis, le bloc d'adaptation de la résolution spatiale ajuste la résolution spatiale pour chaque trame et décide les modes résultants de la nouvelle taille. Finalement, l'encodeur raffine les VMs par le biais d'une fenêtre de recherche de petite taille et réestime les modes de codage des MBs.

4.3.2.3 Blocs invoqués pour l'adaptation du format

Après le décodage des trames dans le domaine pixel et la récupération des méta-informations, l'adaptation de format est implémentée en utilisant le bloc de mappage des VMs pour la transformation des VMs de la syntaxe utilisée en entrée vers la syntaxe utilisée par la sortie, puis il réalise la conversion des modes de MBs pour rencontrer la syntaxe du format en sortie.

4.3.2.4 Blocs invoqués pour l'insertion de logo

L'insertion de logo est implémentée par le bloc d'insertion de logo après le décodage dans le domaine pixel et l'extraction des méta-informations en entrée. Pour minimiser le nombre des MBs affectés par cette insertion, le logo est inséré dans l'endroit désiré, mais il est sujet à une réduction de sa taille horizontale au plus proche multiple de 16 (pour s'aligner sur les MBs)

et une translation au coin supérieur gauche du MB. Le bloc de mappage des VMs est lancé par le bloc d'insertion de logo pour réestimer les VMs des MBs-logo, c.-à-d. les MBs où le logo sera inséré. Les VMs et les modes des MBs non affectés par l'insertion de logo demeurent inchangés. Cependant, les VMs des MBs qui pointaient vers les MBs-logo vont avoir un vecteur (0,0). À la fin, l'encodeur encode la vidéo sans réaliser une réestimation des modes de MBs, ni un raffinement des vecteurs de mouvements.

4.3.2.5 Blocs invoqués pour l'adaptation du débit binaire

L'adaptation du débit binaire de la vidéo compressée passe d'abord par un décodage dans le domaine pixel et la récupération des informations extraites du train de bits en entrée. Ainsi, le train de bits est transmis à l'encodeur où le débit binaire est adapté en ajustant le facteur de quantification, puis en réestimant les modes des MBs pour les MBs qui étaient codés en inter dans le flux vidéo entrant. La réestimation des modes est faite par le biais du bloc de mappage des VMs.

4.3.3 Blocs invoqués pour la réalisation d'un ensemble d'opérations

La Figure 4.2 illustre l'organigramme de réalisation d'une combinaison d'opérations de transcodage vidéo par la nouvelle architecture unifiée. L'ordre des opérations est comme suit:

1. Décodage dans le domaine spatial et stockage des métadonnées (VMs, modes, QP, etc.)
2. Adaptation du taux de trames.
3. Adaptation de la résolution spatiale.
4. Insertion de logo.
5. Adaptation du débit binaire, ré-encodage et contrôle du débit.

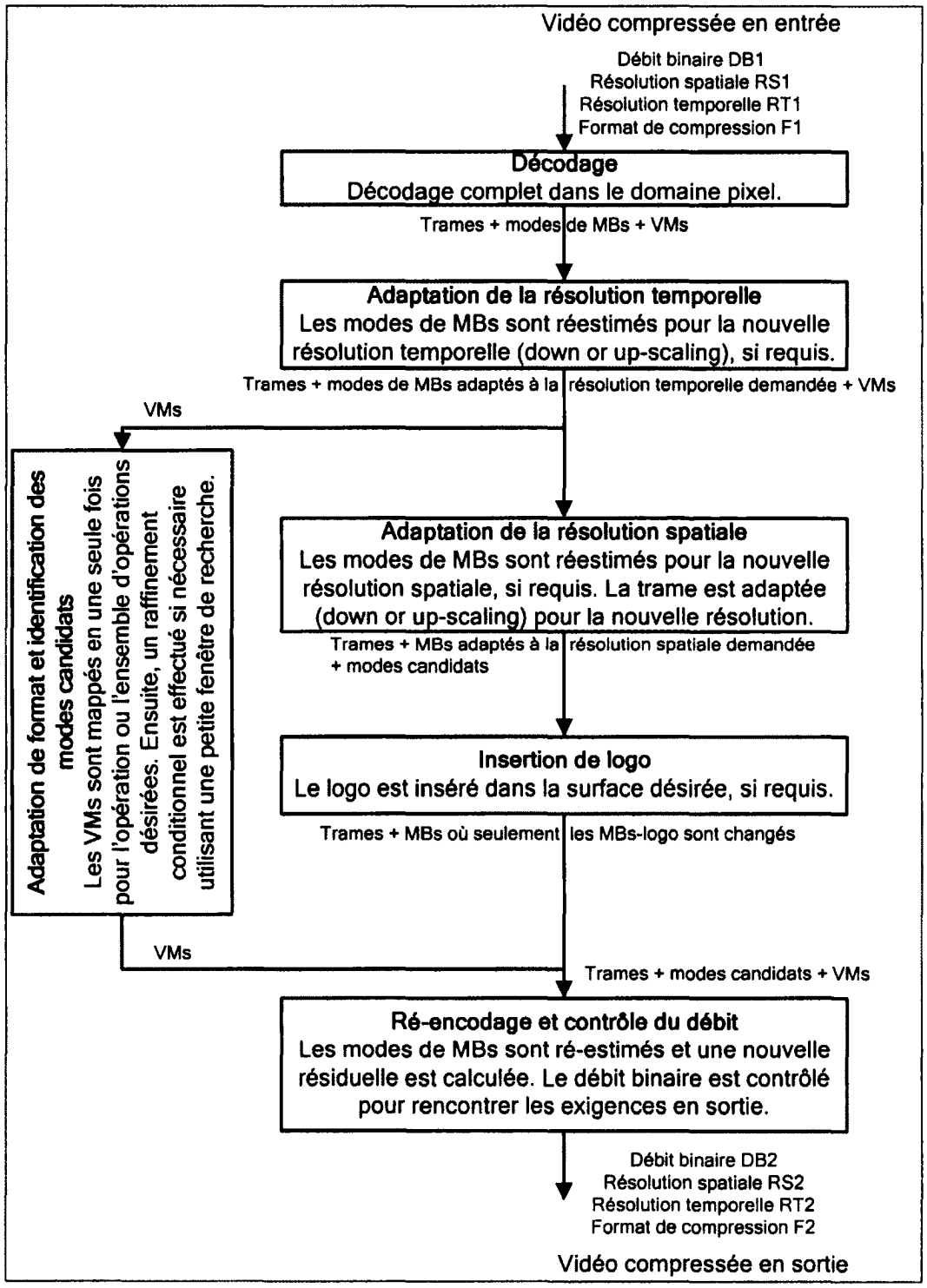

Figure 4.2 Organigramme de l'architecture de transcodage vidéo unifiée.

Cet ordre est le résultat de l'analyse d'un grand nombre d'architectures de transcodage vidéo, et surtout celles qui ont été présentées dans le chapitre 3.

Chaque opération peut être invoquée ou non, selon les caractéristiques désirées en sortie, permettant ainsi d'effectuer n'importe quelle combinaison souhaitée ou d'effectuer une seule opération. Dans le cas où toutes les fonctionnalités seraient demandées, par exemple, l'architecture réalise un décodage, et se sert des trames et des méta-informations pour adapter le taux de trames. Ainsi, un mappage de 1 à 1 permettra d'adapter le taux de trames en utilisant le format de compression en entrée en utilisant l'algorithme FDVS. Les trames et les VMs ainsi obtenus forment l'entrée pour la prochaine étape, soit l'adaptation du format avec adaptation de la résolution spatiale, insertion de logo et adaptation du débit binaire. Un seul mappage des modes et des VMs est effectué pour les quatre opérations. À la fin, la vidéo est réencodée avec un raffinement des VMs pour augmenter la qualité sans affecter le temps de transcodage, tout en contrôlant le débit binaire en sortie.

Si une combinaison qui contient moins d'opérations à réaliser, tout ce qu'il faut faire est de passer les opérations non souhaitées. Par exemple, s'il est demandé d'effectuer une adaptation de la résolution spatiale avec une insertion de logo, le système va décoder le flux video entrant complètement dans le domaine pixel puis réalise un seul mappage des modes et VMs pour les deux opérations. À la fin, le réencodage, le raffinement des VMs et le contrôle du débit binaire sont effectués.

Il est à noter que dans le cas où toutes les fonctionnalités seraient demandées, le mappage des VMs se fait en deux étapes. Dans la première étape, on réalise un mappage de 1 à 1 seulement pour l'adaptation du taux de trames. Pour la deuxième étape, un seul mappage pour le reste des fonctionnalités est effectué. Cependant, si une combinaison d'opération est demandée incluant l'adaptation du taux de trames mais n'incluant pas l'adaptation de format, à ce moment là un seul mappage sera effectué.

Si une seule opération est demandée, par exemple l'adaptation du taux de trames, le système fait un passage du décodage vers l'adaptation du taux de trames puis vers le réencodage, le raffinement et le contrôle du débit binaire.

Les blocs invoqués pour la réalisation d'une combinaison d'opérations dépendent de la combinaison elle-même. Si une opération est demandée dans une combinaison, cela implique

que le ou les blocs nécessaires pour la réalisation de cette opération sont invoqués pour la réalisation de la totalité des opérations dans l'ordre décrit ci-dessus.

4.4 Algorithmes proposés

Afin de démontrer l'efficacité de la nouvelle architecture, nous proposons certains algorithmes pour la réalisation des deux types de scénarios de transcodage vidéo. Dans le premier scénario, une seule opération est implémentée où nous allons implémenter le cas de la réduction du débit binaire puis le cas de la réduction de la résolution spatiale. Les deux opérations sont réalisées pour H.264. Dans le second cas, un ensemble d'opération est réalisé où nous allons implémenter le changement de format (avec ou sans réduction du débit binaire) puis le changement de format avec réduction de la résolution spatiale (avec ou sans réduction du débit binaire). Le changement de format est effectué de H.264 à MPEG-4. Dans ce qui suit, la notation P sera utilisée pour indiquer une trame inter de type P. La notation I quant à elle est utilisée pour les trames intra de type I. Les algorithmes proposés sont implémentés en utilisant le code d'Intel® IPP [18] tel que nous allons le voir dans les chapitres 5 et 6. Pour H.264, les modes de MBs des trames P sont testés selon l'ordre suivant : P16x16, Skip, P8x8, P16x8, P8x16, I16x16 puis I4x4. Pour les modes de MBs des trames I, l'ordre est le suivant : I16x16 puis I4x4. Pour MPEG-4, les modes de MBs des trames P sont testés selon l'ordre suivant : P16x16, Skip, P8x8 puis Intra. Pour les trames I, on n'a que le mode Intra à tester.

Tel que nous allons le voir, IPP ne teste pas tous les modes comme les codes de références et propose l'utilisation de méthodes de prédiction afin de choisir rapidement le meilleur mode.

4.4.1 Adaptation du débit binaire

L'algorithme d'adaptation du débit binaire est implémenté en se basant sur l'analyse des statistiques d'un grand nombre de séquences vidéos transcodées à différents débits binaires. Dans cet algorithme, quoique pour chaque MB en entrée, un mappage vers différents modes en sortie est proposé, cependant on pourrait tester certains modes seulement pour prendre une décision. De plus, cet algorithme propose de tester certains modes inter lorsque le mode en entrée est I16x16 ou I4x4 (voir tableau 4.1). La raison est que si on transcode les MBs I16x16 et I4x4 -dans une trame P- à des MBs intra, cela pourrait générer plus de bits dans le

train de bits sans augmenter nécessairement la qualité. Aussi, il faut noter qu'en plus de l'algorithme proposé, une variante de cet algorithme est introduite, appelée algorithme proposé avec raffinement conditionnel, qui consiste à réaliser un raffinement conditionnel des VMs en réutilisant la précision des VMs entrants. Notre algorithme d'adaptation du débit binaire est décrit comme suit.

Pour les trames intra (I), si le MB est codé en I16x16, il est transcodé en I16x16. S'il est encodé en I4x4, alors I16x16 puis I4x4 sont testés.

Pour les trames inter (P), le tableau 4.1 illustre l'algorithme de mappage utilisé.

a. Si le MB en entrée est encodé en skip, les modes skip et inter16x16 (P16x16) sont testés en sortie. Pour tester P16x16, le VM issu du mode skip en entrée est utilisé. Le mode skip est toujours testé en utilisant le VM prédit -PMV (*median Predicted Motion Vector*).-
b. Si le MB est encodé en P16x16, skip et P16x16 sont testés. Pour tester P16x16, le VM en entrée est utilisé.
c. Si le MB en entrée est encodé en P16x8, skip, P16x16 et P16x8 sont testés. La raison pour laquelle skip et P16x16 sont testés est le fait que l'encodeur utilise plus de modes de taille large pour les débits faibles. P16x16 est testé en calculant la moyenne des deux VMs P16x8. Pour P16x8, pour chaque bloc 16x8, le VM en entrée est employé.
d. Si le MB en entrée est encodé en P8x16, skip, P16x16 et P8x16 sont testés. P16x16 est testé en calculant la moyenne des deux VMs P8x16. Pour P8x16, chaque bloc 8x16 est testé en réutilisant le VM entrant.
e. Si le MB est encodé en P8x8, tous les modes sont testés en adaptant les VMs de P8x8, vu que l'encodeur emploie des MBs de grande taille lorsque le débit binaire est faible. D'abord, si un bloc 8x8 contient une partition moins que 8x8 (8x4, 4x8 ou 4x4), ce dernier est transformé en bloc 8x8. P16x16 est testé en calculant la médiane ou la moyenne des 4 VMs P8x8. Pour P16x8 et P8x16, chaque bloc est testé avec la moyenne des deux VMs correspondants. Finalement, chaque bloc 8x8 en sortie est testé en utilisant le VM du bloc correspondant en entrée.
f. Si le MB est I16x16, skip, P16x16 et I16x16 sont testés en sortie. Les modes inter sont testés parce qu'à faible débit, il y a une grande possibilité pour que les MBs intra se transforment en MBs inter. P16x16 est testé dans ce cas en utilisant le VM (0, 0).

g. Si le MB est I4x4, skip, P16x16, I16x16 et I4x4 sont testés en sortie. P16x16 est testé dans ce cas en utilisant le VM (0, 0), ensuite I16x16 et I4x4 sont testés.

Il est à noter que lorsqu'on teste une partition de type 8x8, les sous modes (8x4, 4x8 et 4x4) ne sont pas testés car ils n'apportent pas une qualité significativement meilleure mais augmentent les calculs.

Tableau 4.1 Algorithme proposé pour l'adaptation du débit binaire en H.264

	Entrée	**Sortie**
a	Skip	Tester (Skip + P16x16)
b	P16x16	Tester (Skip + P16x16)
c	P16x8	Tester (Skip + P16x16 + P16x8)
d	P8x16	Tester (Skip + P16x16 + P8x16)
e	P8x8	Tester (Skip + P16x16 + P16x8 + P8x16 + P8x8)
f	I16x16	Tester (Skip + P16x16 + I16x16)
g	I4x4	Tester (Skip + P16x16 + I16x16 + I4x4)

Pour le raffinement des VMs finaux au quart de pixel, deux approches sont employées en utilisant une fenêtre de recherche de taille ±1 (¼ de pixel) pour a, b, c, d, e, f et g. Dans la première approche (algorithme proposé), tous les VMs finaux sont raffinés. Pour la seconde (algorithme proposé avec raffinement conditionnel), la précision des VMs entrants est réutilisée sauf pour le cas où le meilleur VM est le VM prédit. Comme on le verra, cela a un effet remarquable sur la rapidité du transcodeur en maintenant une qualité similaire à la première approche.

4.4.2 Adaptation de la résolution spatiale

Dans ce qui suit, nous proposons un algorithme pour la réduction de la résolution spatiale avec un facteur de deux. Cet algorithme a été développé en se basant sur l'analyse des statistiques du transcodage d'un grand nombre de séquences vidéos avec le même débit ou à

des débits binaires réduits. Contrairement aux algorithmes présentés dans la littérature, notre algorithme propose un mappage direct pour les cas de quatre modes Skip ou P16x16. De plus, une variante de l'algorithme proposé qui consiste à réaliser un raffinement conditionnel des VMs est présentée. Les algorithmes d'adaptation de la résolution spatiale pour les autres standards pourraient être implémentés en suivant la même approche. La Figure 4.3 illustre le problème de mappage des modes et des VMs.

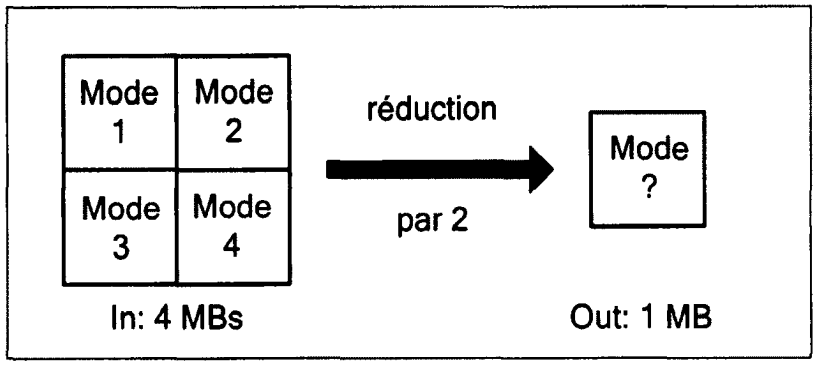

Figure 4.3 Mappage des modes pour la réduction de la résolution spatiale avec un facteur de 2.

Pour les trames intra, I16x16 et I4x4 sont testés.

Pour les trames inter, l'algorithme de mappage est illustré par le tableau 4.2.

a. Si les quatre MBs sont skip, skip et P16x16 sont testés. Le VM de P16x16 est calculé par la médiane ou la moyenne des quatre VMs skip entrants. Dans nos simulations, la moyenne est utilisée. Le mode skip est toujours testé avec le VM prédit (car c'est le vecteur associé au mode skip en H.264, le format de sortie). Les VMs en entrée sont mis à l'échelle (division par 2) avant d'être utilisés.

b. Si les quatre MBs sont P16x16, skip et P16x16 sont testés. Le VM de P16x16 est calculé par la médiane des quatre VMs P16x16 entrants. Les VMs finaux sont divisés par 2 avant d'être utilisés.

c. Si les quatre MBs sont inter, à l'exception des cas où on a quatre skip ou quatre P16x16, tous les modes sont testés en sortie. D'abord, chaque MB en entrée est transformé en P16x16. Pour tester P16x16, la moyenne des quatre P16x16 est employée. Pour P16x8, chaque bloc est formé en utilisant la moyenne des MBs correspondants, soit le MB n° 1 et le MB n° 2 pour le bloc 16x8 de haut et le MB n° 3 et le MB n°4 pour le deuxième bloc 16x8. De même, pour P8x16, chaque bloc est formé en utilisant la moyenne des

MBs correspondants, soit le MB n° 1 et le MB n° 3 pour le bloc 8x16 de gauche et le MB n° 2 et le MB n° 4 pour le deuxième bloc 8x16. Pour P8x8, chaque bloc 8x8, bloc 1, 2, 3 ou 4 est testé en utilisant le VM du MB correspondant en entrée. Les VMs en entrée sont mis à l'échelle avant d'être utilisés.

d. S'il y a un MB I16x16 ou I4x4 parmi les quatre MBs en entrée, skip, P16x16, P8x8, I16x16 et I4x4 sont testés. D'abord, chaque MB en entrée est transformé en P16x16. I16x16 et I4x4 sont transformés en employant le VM (0, 0). P16x16 est testé en calculant la moyenne des quatre P16x16s. Pour P8x8, chaque bloc 8x8, bloc 1, 2, 3 ou 4 est testé en utilisant le VM du MB correspondant en entrée. Les VMs en entrée sont mis à l'échelle (division par 2) avant d'être utilisés. Après le test des modes inter, I16x16 puis I4x4 sont testés.

Il est à noter que lorsqu'on teste une partition de type 8x8, les sous modes (8x4, 4x8 et 4x4) ne sont pas testés car ils n'apportent pas une qualité significativement meilleure mais augmentent les calculs.

Pour le raffinement des VMs finaux au quart de pixel, trois approches sont employées en utilisant une fenêtre de recherche de taille ±1 (¼ de pixel) pour a, b, c, et d. Dans la première approche (algorithme proposé), tous les VMs finaux sont raffinés. Pour la seconde (algorithme proposé avec raffinement conditionnel), la précision des VMs entrants est réutilisée sauf pour le cas où le meilleur VM est le VM prédit. Pour la troisième approche, aucun raffinement n'est effectué (algorithme proposé sans raffinement).

Tableau 4.2 Algorithme proposé pour l'adaptation de la résolution spatiale en H.264

	Entrée (4 MBs)	**Sortie (1 MB)**
a	4 Skip	Tester (Skip + P16x16)
b	4 P16x16	Tester (Skip + P16x16)
c	4 Inter (*modes inter hétérogènes exceptés 4 Skip ou 4 P16x16*)	Tester (Skip + P16x16 + P16x8 + P8x16 + P8x8)
d	nb_mb_intra4x4 > 1 ou nb_mb_intra16x16 > 1	Tester (Skip + P16x16 + P8x8+ I16x16 + I4x4)

4.4.3 Adaptation du format

L'algorithme d'adaptation du format en H.264 *Baseline Profile* et MPEG-4 *Visual Simple Profile* est décrit comme suit. Pour les transcodages entre les autres normes, l'approche serait la même.

Pour les trames intra (I), si le MB est codé en I16x16 ou en I4x4, il est transcodé en intra, car MPEG-4 supporte seulement le mode I16x16.

Pour les trames inter, le tableau 4.3 illustre l'algorithme de mappage utilisé.

a. Si le MB en entrée est encodé en skip, les modes skip et P16x16 sont testés en sortie. Pour tester P16x16, le VM issu du mode skip en entrée est utilisé. En MPEG-4, le mode skip est toujours testé en utilisant le VM (0, 0).
b. Si le MB est encodé en P16x16, skip et P16x16 sont testés. Pour tester P16x16, le VM en entrée est utilisé.
c. Si le MB en entrée est encodé en P16x8 ou P8x16, skip et P16x16 sont testés. Pour tester P16x16, la moyenne des deux VMs P16x8 ou P8x16 est utilisée.
d. Si le MB est encodé en P8x8, skip et P16x16, P8x8 sont testés. Le VM P16x16 est calculé par la médiane ou la moyenne des quatre VMs P8x8 en entrée. Finalement, chaque bloc 8x8 en sortie est testé en utilisant le VM du bloc correspondant en entrée.
e. Si le MB est I16x16 ou I4x4, les modes skip, P16x16 et intra sont testés. P16x16 est testé en employant le VM (0, 0). Ensuite, intra est testé.

Pour le raffinement des VMs finaux au demi-pixel, deux approches sont employées en utilisant une fenêtre de recherche de taille ±1 (½ pixel) pour a, b, c, d, et e. Dans la première approche (algorithme proposé), tous les VMs finaux sont raffinés. Pour la seconde (algorithme proposé avec raffinement conditionnel), la précision des VMs entrants est réutilisée sauf pour le cas où le meilleur VM est le VM prédit. Comme on le verra, cela a un effet remarquable sur la rapidité du transcodeur en maintenant une qualité similaire à la première approche.

Tableau 4.3 Algorithme proposé pour l'adaptation de format de H.264 BP à MPEG-4 VSP

	Entrée (H.264)	**Sortie (MPEG-4)**
a	Skip	Tester (Skip + P16x16)
b	P16x16	Tester (Skip + P16x16)
c	P16x8 ou P8x16	Tester (Skip + P16x16)
d	P8x8	Tester (Skip + P16x16 + P8x8)
e	I16x16 ou I4x4	Tester (Skip + P16x16 + Intra)

4.4.4 Adaptation du format avec adaptation de la résolution spatiale

L'algorithme d'adaptation du format en H.264 *Baseline Profile* et MPEG-4 *Visual Simple Profile* avec réduction de la résolution spatiale par un facteur de 2 est décrit comme suit.

Pour les trames intra, les trames sont transcodées en intra.

Pour les trames inter, l'algorithme de mappage est illustré par le tableau 4.4.

a. Si les quatre MBs sont skip, skip et P16x16 sont testés. Le VM de P16x16 est calculé par la médiane ou la moyenne des quatre VMs skip entrants. Le mode skip est toujours testé avec le VM (0,0) (car c'est le vecteur associé au mode skip en MPEG-4, le format de sortie). Les VMs en entrée sont mis à l'échelle (division par 2) avant d'être utilisés.

b. Si les quatre MBs sont P16x16, skip et P16x16 sont testés. Le VM de P16x16 est calculé par la médiane des quatre VMs P16x16 entrants. Les VMs finaux sont divisés par 2 avant d'être utilisés.

c. Si les quatre MBs sont inter, à l'exception des cas où on a quatre skip ou quatre P16x16, tous les modes inter sont testés en sortie. D'abord, chaque MB en entrée est transformé en P16x16. Pour tester P16x16, la moyenne des quatre P16x16 est employée. Pour P8x8, chaque bloc 8x8, bloc 1, 2, 3 ou 4 est testé en utilisant le VM du MB correspondant en entrée. Les VMs en entrée sont mis à l'échelle avant d'être utilisés.

d. S'il y a un MB I16x16 ou I4x4 parmi les quatre MBs en entrée, skip, P16x16, P8x8, intra sont testés. D'abord, chaque MB en entrée est transformé en P16x16. I16x16 et I4x4 sont transformés en employant le VM (0, 0). P16x16 est testé en calculant la moyenne des quatre P16x16s. Pour P8x8, chaque bloc 8x8, bloc 1, 2, 3 ou 4 est testé en

utilisant le VM du MB correspondant en entrée. Les VMs en entrée sont mis à l'échelle (division par 2) avant d'être utilisés. Après le test des modes inter, le mode intra est testé.

Pour le raffinement des VMs finaux au demi-pixel, une seule approche est employée en utilisant une fenêtre de recherche de taille ±1 (½ pixel) pour a, b, c, et d. Dans cette approche (algorithme proposé), tous les VMs finaux sont raffinés.

Il est important de noter que cet algorithme est similaire à l'algorithme décrit pour la réduction spatiale à l'exception que chacun d'eux teste des modes spécifiques en fonction du format de sortie. En effet, les modes P16x8 et P8x16 existent seulement en H.264. On peut remarquer plus facilement cette différence dans les tableaux 4.1 et 4.3. Donc de manière générale, on applique le même algorithme mais adapté aux formats spécifiques d'entrée et de sortie.

Tableau 4.4 Algorithme proposé pour l'adaptation de format de H.264 BP à MPEG-4 VSP avec adaptation de la résolution spatiale

	Entrée (H.264) **4 MBs**	**Sortie (MPEG-4)** **1 MB**
a	4 Skip	Tester (Skip + P16x16)
b	4 P16x16	Tester (Skip + P16x16)
c	4 Inter (*modes inter hétérogènes exceptés 4 Skip ou 4 P16x16*)	Tester (Skip + P16x16 + P8x8)
d	nb_mb_intra4x4 > 1 ou nb_mb_intra16x16 > 1	Tester (Skip + P16x16 + P8x8+ Intra)

4.5 Conclusion

Nous avons commencé par formuler le problème des architectures de transcodage vidéo qui existent dans le marché du multimédia actuellement. Une architecture générique capable de réaliser n'importe quelle combinaison d'opérations, tout en générant la meilleure qualité possible dans le moindre temps de transcodage est indispensable. En plus, cette architecture doit être extensible pour les cas futurs en minimisant les efforts et les coûts de développement et maintenance. Nous avons présenté une architecture de transcodage vidéo, réalisée dans le domaine pixel pour flexibilité et extensibilité aux cas futurs, une très haute qualité possible et ayant un temps de calcul très bas en exploitant au maximum les méta-informations récupérées du train de bits en entrée. Ensuite, nous avons proposé les algorithmes développés pour démontrer l'efficacité de la nouvelle architecture pour les deux scénarios possibles : une seule opération ou un ensemble d'opérations.

CHAPITRE 5

ARCHITECTURE LOGICIELLE

5.1 Introduction

Après avoir conçu la nouvelle architecture unifiée de transcodage vidéo pour pouvoir effectuer toutes sortes d'opérations de transcodage, nous présentons dans ce chapitre l'architecture logicielle qui sera utilisée dans les expériences et simulations. D'abord, nous présentons la librairie Intel® *Integrated Performance Primitives* (Intel® IPP) permettant de réaliser un grand nombre de fonctionnalités de base de transcodage vidéo. Ensuite, nous détaillons les modifications qui ont été apportées à Intel® IPP afin d'implémenter notre nouvelle architecture ainsi que les interactions avec Intel® IPP. À la fin, nous concluons ce chapitre.

5.2 Architecture logicielle

5.2.1 Intel® Integrated Performance Primitives (Intel® IPP)

Intel® IPP [18] est une librairie logicielle utilisée dans une multitude d'applications. Du domaine scientifique, en passant par le multimédia et le web (traitement d'images, de la vidéo, etc.) jusqu'aux systèmes embarqués et les télécommunications, Intel® IPP fournit une plateforme de très haute performance sous un ensemble d'API écrits en C/C++ afin de supporter un grand nombre de systèmes d'exploitation : Windows, Linux et Mac OS X.

La Figure 5.1 [17] illustre l'architecture haut niveau d'Intel® IPP. Les API offrent aux programmeurs et chercheurs la possibilité de développer une variété d'applications dans différents domaines. Intel® fournit un code illustratif gratuitement (*free code sample*) pour démontrer certaines fonctionnalités qui peuvent être réalisées à l'aide de sa plateforme. Une fois ce code compilé, un ensemble d'exécutables optimisés sera généré selon le système d'exploitation et les caractéristiques de la machine cible (32 ou 64 bits, multi-cœurs, etc.).

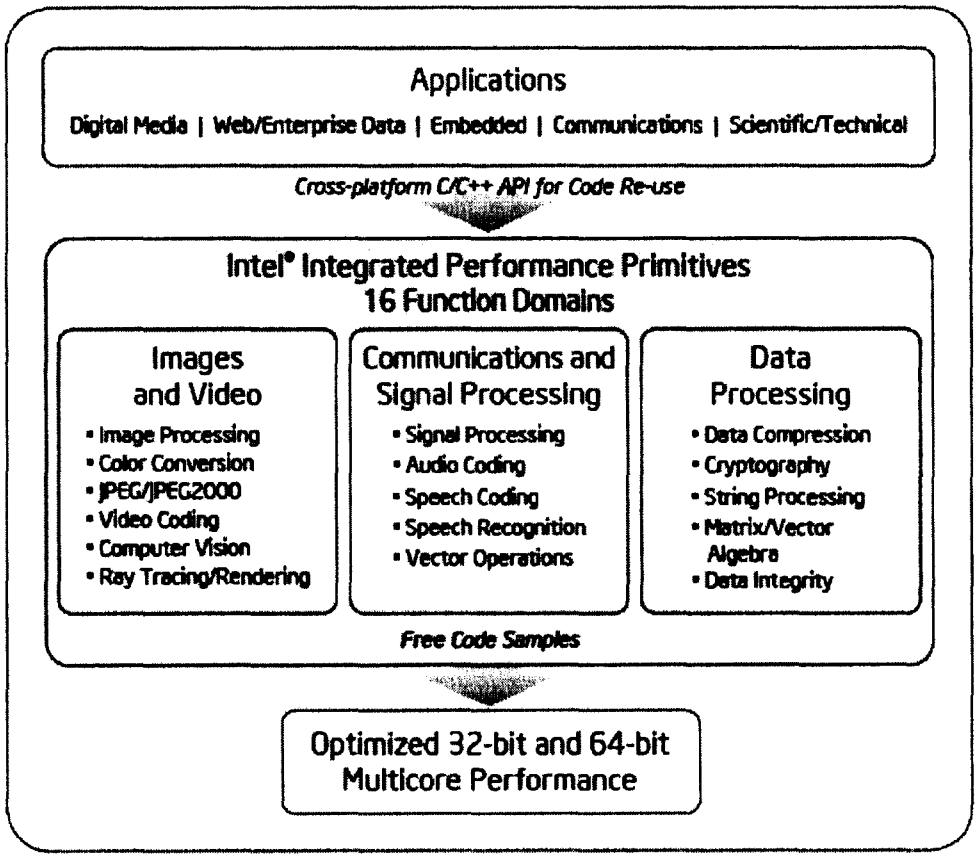

Figure 5.1 Architecture d'Intel® IPP.

Tirée d'Intel® IPP (2010, en ligne)

Plusieurs raisons nous ont poussées à choisir Intel® IPP pour implémenter notre projet, parmi lesquelles :

- Une plateforme optimisée. Le code final est compilé selon le système d'exploitation et le processeur cibles. Pour notre cas, les résultats obtenus seront plus réalistes et plus proches des transcodeurs temps réels utilisés en industrie.
- Elle supporte différentes sortes de processeurs d'Intel® -du processeur x86 jusqu'aux multi-cœurs- et aussi les processeurs compatibles comme ceux d'AMD®.
- Elle supporte un grand nombre de fonctions : traitement d'images 2-D et 3-D, codage et transcodage vidéo, cryptographie, reconnaissance de la voix, etc.
- Les standards vidéos les plus utilisés y sont supportés : MPEG-2, H.263, MPEG-4, H.264, et VC1.

- Une plateforme développée par la même compagnie, donc la même logique est suivie partout, ce qui laisse le développeur familier avec cette plateforme après une certaine période.
- Les fonctionnalités de bases sont bien documentées, ce qui permet aux développeurs de les utiliser efficacement.

Dans ce mémoire, nous avons utilisé la version 5.3 d'Intel® IPP. Cette version inclut un encodeur et un décodeur pour la majorité des standards de compression utilisés dans le marché du multimédia, notamment MPEG-2, H.263, MPEG-4, H.264, et VC1. En plus, un transcodeur vidéo cascadé est développé en se basant sur ces standards, donc toute sorte de transcodage -de MPEG-2 à VC-1 ou de H.263 à H.264, etc.- sera possible. La Figure 5.2 illustre le schéma du transcodeur d'Intel® IPP. Ce dernier appelle le décodeur selon le format de compression de la vidéo pour effectuer un décodage complet dans le domaine pixel. Ensuite, il réalise les opérations requises comme l'adaptation de la résolution spatiale. À la fin, il appelle l'encodeur vidéo selon le format de compression désiré pour ré-encoder la vidéo.

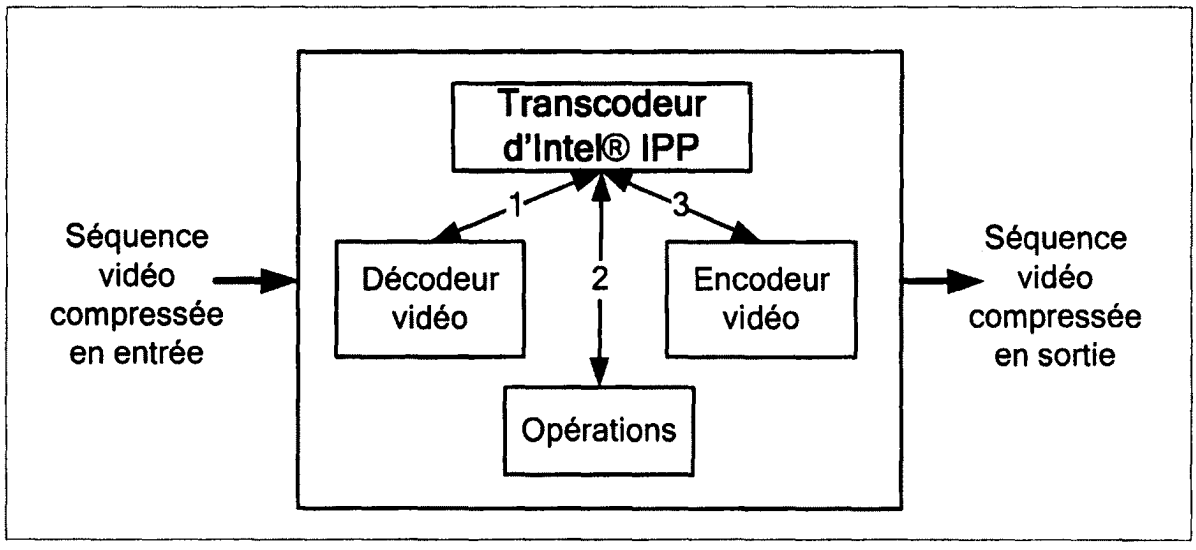

Figure 5.2 Schéma du transcodeur d'Intel® IPP.

5.2.2 Implémentation de l'architecture unifiée

Afin de prouver l'utilité et l'efficacité de la nouvelle architecture unifiée de transcodage vidéo proposée dans le chapitre 4, nous avons bâti notre solution sur la plateforme Intel® IPP. Cette nouvelle plateforme, appelée UVECT (*Unified Video Engine for Coding and Transcoding*) se veut être une plateforme générique pour répondre à tous les besoins en matière de codage et de transcodage vidéo telle que discutée dans le chapitre 4.

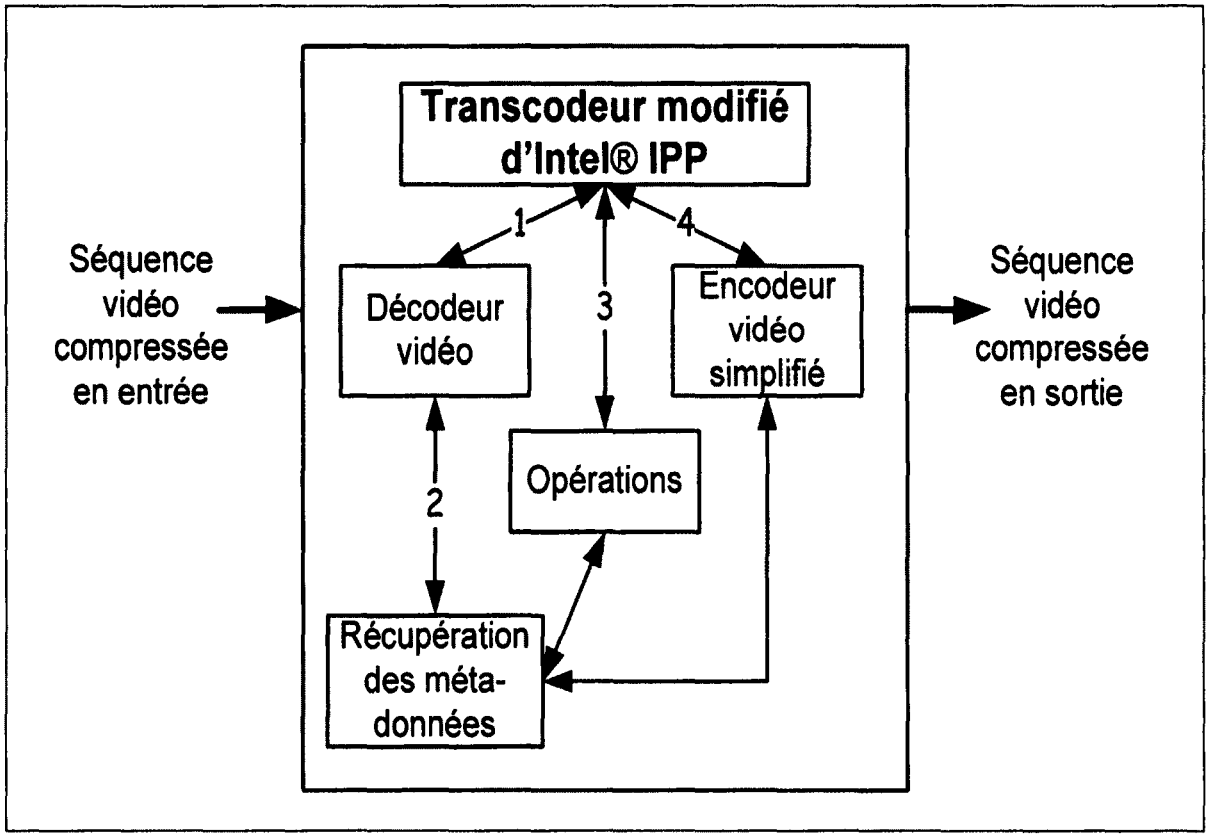

Figure 5.3 Schéma de la plateforme uVect.

La Figure 5.2 illustre le schéma haut niveau de la plateforme UVECT. Un nouveau module, appelé module de récupération des métadonnées a été rajouté. Ce dernier est utilisé pour stocker les métadonnées extraites du décodeur. Le transcodeur vidéo a été modifié afin de permettre plus d'interaction et de contrôle sur la totalité de la plateforme. Le transcodeur va donc demander à ce module de récupérer les métadonnées du décodeur, ensuite il va réaliser les opérations demandées en tenant compte des métadonnées récupérées. À la fin, le

transcodeur demande à l'encodeur simplifié de ré-encoder la séquence avec les paramètres désirés en sortie en évitant une nouvelle estimation de mouvement.

Le module de récupération des métadonnées[5] est montré par la Figure 5.4. Ce module comprend un ensemble de classes où la classe MetaDataMap est la classe mère. Cette classe est instanciée pour chaque slice afin de stocker toutes les données de ses MBs. Dépendamment des caractéristiques de la vidéo, on peut avoir jusqu'à concurrence de trois objets de la classe MetaDataMap, c.-à-d. une pour chaque plan Y, Cb et Cr. La classe MacrBlockTree est une classe agrégée de MetaDataMap où chaque objet représente un MB de la slice courante. Chaque objet MetaDataMap contient autant d'objets MacrBlockTree qu'il y a par slice. L'objet MB représente la structure d'un MB telle que définie par les standards de compression et contient vingt-et-un objets de la classe Block. Pour H.263 profil 0, on peut avoir seulement une seule partition 16x16. Ce sera le Block n° 1 qui va contenir les données pour cette partition. Pour MPEG-4 par exemple, on peut avoir une seule partition 16x16 ou quatre partitions 8x8, donc dans ce cas nous allons avoir cinq objets Block. Le premier va indiquer qu'il s'agit d'une partition 8x8 et ne contiendra aucune donnée. Les objets Block n° 2, 3, 4 et 5 vont stocker les métadonnées pour les partitions 8x8 n° 1, 2, 3 et 4 respectivement. Pour H.264, si on a un MB avec quatre partitions 8x8 et où chaque partition est subdivisée en quatre partitions 4x4, alors nous allons avoir vingt-et-un objets Block. Le premier objet indiquera que le bloc 16x16 est divisé en 4 partitions 8x8, les 4 objets suivants que chaque partition 8x8 est subdivisée en partitions 4x4, et les données seront stockées dans les objets n° 5 jusqu'à 21 (voir Figures 5.5 et 5.6). Cette méthode hiérarchique est utilisée peu importe le format et le partitionnement. Pour H.264 il y a un maximum de 21 objets mais il peut y en avoir moins selon le partitionnement choisi. La taille de chaque objet Block ainsi que la partition (16x16, 8x8, etc.) est connue à travers l'attribut de type Partition. Le type du Block (inter, intra_dc, etc.) est déterminé par l'attribut

[5] Module développé par Steven Pigeon, professionnel de recherche à l'ÉTS, et Simon Descarries de Vantrix Inc.

CodingMode. En plus, pour chaque objet Block, on peut avoir un ou deux vecteurs de mouvement dépendamment si la trame est P ou B.

Pour plus de détails sur l'ensemble des classes, voir Annexe I.

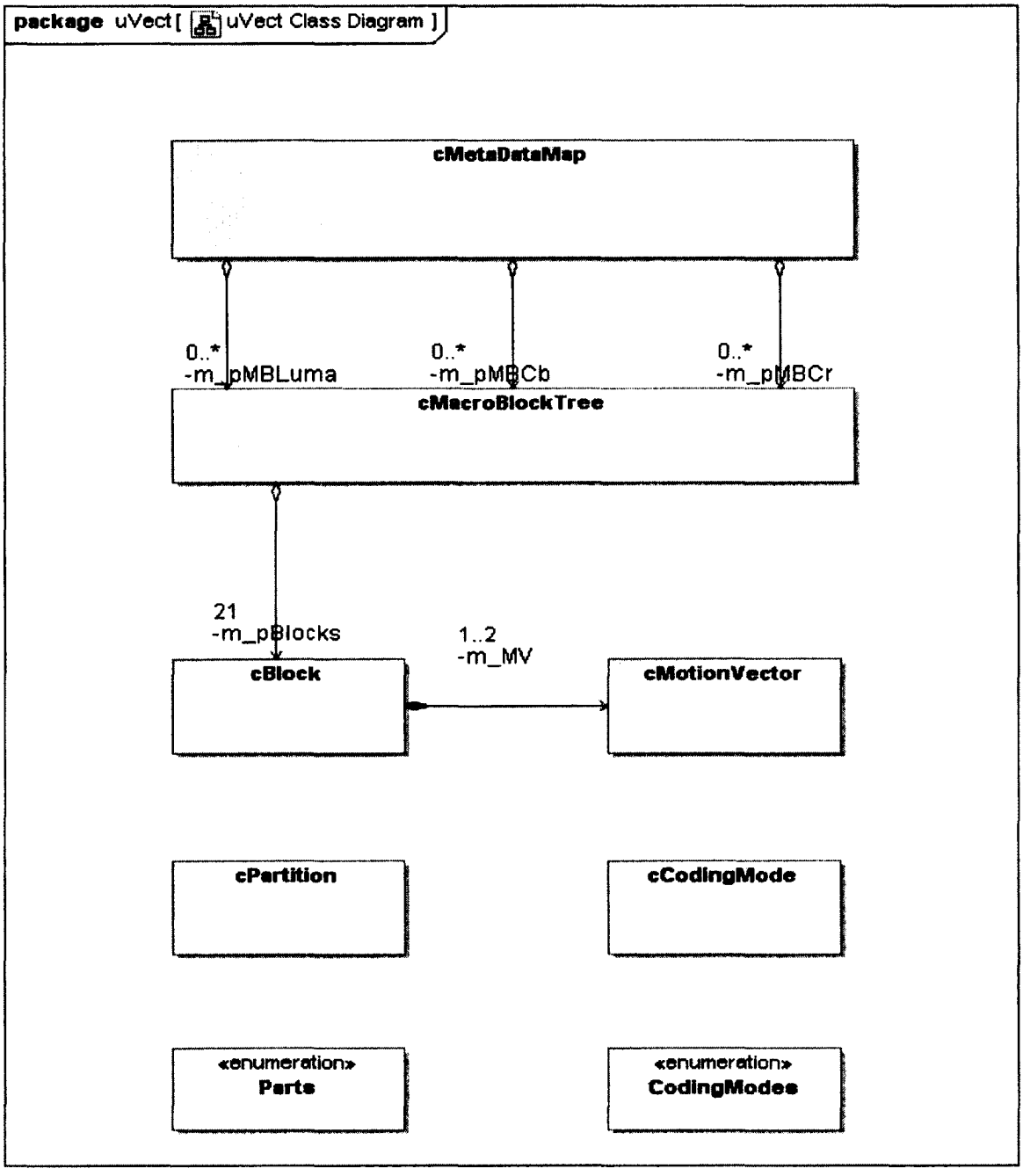

Figure 5.4 Bloc de récupération des métadonnées dans la plateforme uVect - diagramme de classes.

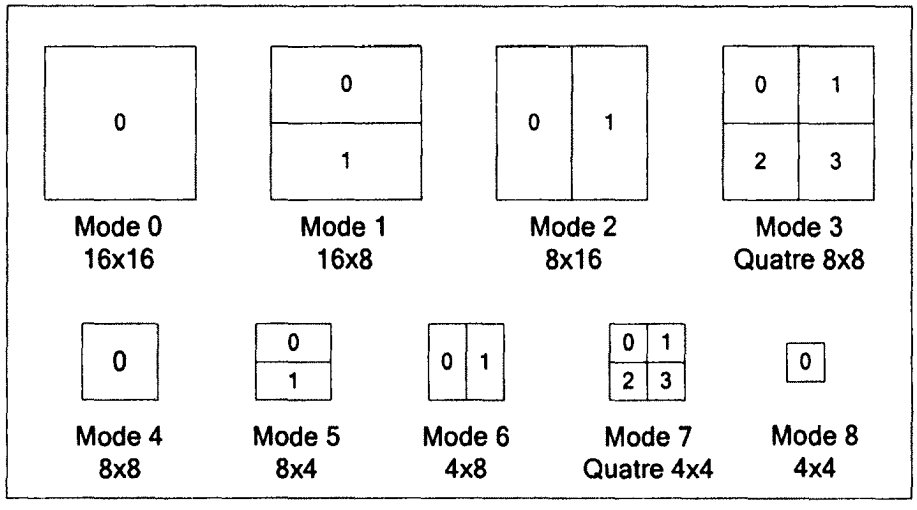

Figure 5.5 Représentation interne des modes de MBs dans uVect[6].

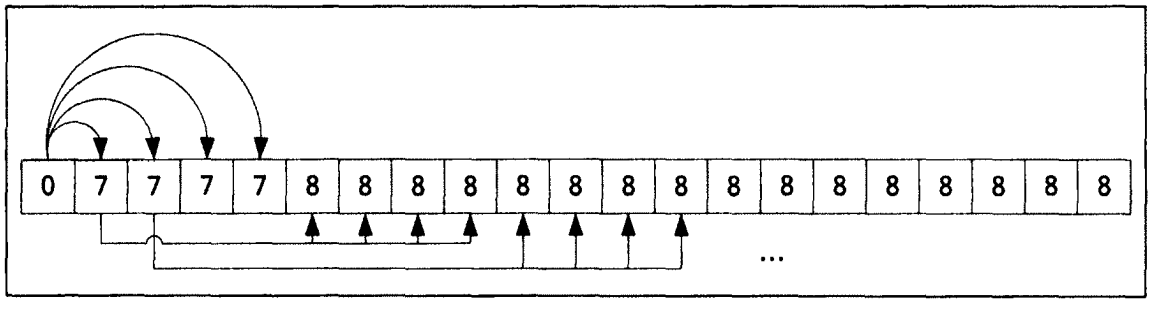

Figure 5.6 Exemple d'un MB avec seize partitions 4x4.

[6] Module développé par Steven Pigeon, professionnel de recherche à l'ÉTS, et Simon Descarries de Vantrix Inc.

5.3 Conclusion

Ce chapitre a donné un bref survol sur Intel® IPP, la librairie logicielle utilisée pour effectuer les tests et simulations sur l'architecture unifiée de transcodage vidéo qui a été conçue dans le cadre de ce mémoire. Intel® IPP nous permet de réaliser un transcodage en cascade, sans réutilisation des métadonnées récupérées au moment du transcodage. De ce fait, nous avons montré les changements requis au niveau architectural, pour développer une architecture capable de réutiliser les métadonnées extraites du décodeur.

CHAPITRE 6

SIMULATIONS ET RÉSULTATS

6.1 Introduction

Ce chapitre présente les simulations qui ont été réalisées pour démontrer l'efficacité de la nouvelle architecture unifiée de transcodage vidéo.

6.2 Méthodologie de simulation

Pour valider l'utilité de la nouvelle architecture unifiée de transcodage vidéo, nous avons implémenté un grand nombre de fonctionnalités de transcodage en faisant appel à différents algorithmes pour chaque cas d'utilisation. Les fonctionnalités de transcodage que nous avons réalisées sont :

1. Adaptation du débit binaire
2. Adaptation de la résolution spatiale
3. Adaptation du format
4. Adaptation du format avec adaptation de la résolution spatiale.

Malgré que nous n'ayons pas implémenté la totalité des opérations dû à la complexité de développement énorme que cela aurait demandé, ce que nous avons implémenté permet de démontrer que notre système unifié donne des résultats très impressionnants en ce qui concerne la qualité de la vidéo et des gains en vitesse (par exemple, il est évident que l'architecture supporte l'insertion de logos et il est donc inutile de le démontrer expérimentalement). La réalisation des quatre fonctionnalités proposées n'a jamais été démontrée dans la littérature. Ainsi, nous proposons la réalisation de trois cas où une seule opération est demandée. Dans le quatrième cas, nous proposons la réalisation d'un ensemble d'opérations. De plus, le support du tout nouveau standard H.264 avec toutes ses nouveautés est au cœur de notre système unifié.

Pour chacune des quatre fonctionnalités, nous avons implémenté les algorithmes décrits dans les sections 4.4.1 à 4.4.4. De plus, nous avons implémenté la méthode des statistiques [14; 15; 26; 27; 38] qui consiste à passer les modes de MBs seulement sans les VMs. Les résultats obtenus, soient le gain en vitesse de transcodage en unités et la qualité en décibels, sont comparés pour chaque méthode par rapport à la méthode cascade où une ré-estimation de mouvement est effectuée. Pour la qualité de la vidéo, la différence est montrée dans les tableaux par rapport à la méthode cascade. Pour le temps de transcodage, le temps présenté dans les tableaux est exprimé en unités et non en seconde. Par exemple, si le temps de transcodage pour la méthode cascade est de 2 secondes et le temps pour la méthode des statistiques est de 1,6 secondes, dans ce cas la méthode cascade est notre référence et va avoir une unité tandis que la méthode des statistiques va avoir 2 / 1,6 = 1.25. Donc le gain est 1.25 (25% plus rapide).

Différentes séquences vidéo du groupe *Video Quality Experts Group* (VQEG) de types CIF (352x288 pixels) et QCIF (176x144 pixels) avec des nombres de trames de 90 à 300 trames ont été employées lors des simulations. Ces séquences représentent plusieurs types de mouvements (mouvement lent, moyen et rapide) afin de tester notre architecture dans des cas extrêmes et réalistes en réalisant le transcodage avec des débits binaires variables allant de 512 kb/s à 32 kb/s.
Pour des fins de simplicité, nous avons utilisé le profil baseline de H.264 pour le premier encodage. Ainsi, le patron utilisé dans nos simulations est un patron IPPP... sans trames bidirectionnelles (trames B), où l'on encode une trame intra (trame I) chaque 3,3 secondes (100 trames).

Intel® IPP supporte plusieurs algorithmes d'estimation de mouvement. Dans le cadre de ce projet, les séquences vidéo sont encodées et transcodées avec l'algorithme EPZS (numéro 3 dans IPP H.264) pour H.263 et l'algorithme logarithmique (numéro 4 dans IPP MPEG-4) pour MPEG-4. Il est à noter qu'IPP n'utilise pas le RDO pour décider le meilleur mode, mais un algorithme de prédiction basé sur des seuils prédéfinis pour l'inter et/ou l'intra.

6.2.1 Adaptation du débit binaire

La réduction du débit binaire a été implémentée dans notre projet par un facteur arbitraire de 1.33, 2, 2.67 et 4. Les tableaux 6.1 à 6.4 représentent les résultats expérimentaux de la réduction du débit binaire en H.264. Trois méthodes ont été utilisées : la méthode des statistiques, la méthode proposée et la méthode proposée avec raffinement conditionnel. Dans la méthode des statistiques [38], seule la décision de mode est transmise au ré-encodeur. Un raffinement de ±3 est utilisé pour cette méthode. Telle que décrite dans la section 4.4.1, la méthode proposée réemploie les modes de MBs, VMs et le reste des métadonnées. Les VMs choisis sont raffinés en utilisant une petite fenêtre de recherche de ±1. La méthode proposée avec raffinement conditionnel est dérivée de la méthode proposée où les VMs choisis sont raffinés si et seulement si le présent VM évalué diffère du VM prédit.

Les séquences CIF ont été encodées à 512 et 256 Kb/s respectivement avec une trame intra chaque 100 trames et étaient transcodées à différents débits binaires. Les tableaux 6.1 et 6.2 illustrent les résultats obtenus. De même, nous avons encodé les séquences QCIF à 256 et 128 kb/s respectivement et nous avons transcodé ces séquences à des débits binaires allant de 192 jusqu'à 32 kb/s. Les résultats obtenus sont illustrés par les tableaux 6.3 et 6.4.

En analysant les quatre tableaux, on peut constater que la méthode proposée est 1.7 fois plus rapide que la méthode cascadée dans le pire scénario et presque plus de 2 fois plus rapide en général. La qualité obtenue est la même aux débits binaires élevés et -0.45 dB dans le pire cas si le débit est trop faible. Pour la méthode proposée avec raffinement conditionnel, elle est plus rapide plus que la méthode proposée et la qualité résultante est tout à fait acceptable (bien qu'aux faibles débits binaires où la pire dégradation était -0.81 dB). Ainsi, un compromis devrait être entre la vitesse et la qualité.

Il est à noter que le contrôle du débit binaire pour chaque méthode n'a pas été présenté dans les tableaux car ce dernier a été exécuté par l'encodeur et aucun algorithme de contrôle n'a été proposé.

Tableau 6.1 Résultats de l'adaptation du débit binaire utilisant des séquences CIF@512 kb/s et transcodées à 384, 256, 192 et 128 kb/s

Vidéos CIF		512 à 384 kb/s				512 à 256 kb/s				512 à 192 kb/s				512 à 128 kb/s			
		Cas-cade	Stati-stics	Prop-osée	Raff. Cond.	Cas-cade	Stati-stics	Prop-osée	Raff. Cond.	Cas-cade	Stati-stics	Prop-osée	Raff. Cond.	Cas-cade	Stati-stics	Prop-osée	Raff. Cond.
container	**G. en V.**	1	1.6	1.87	1.99	1	1.53	1.81	1.82	1	1.44	1.78	1.8	1	1.25	1.69	1.68
300 frames	**PSNR (dB)**	41.71	0	-0.16	-0.15	38.35	-0.02	-0.17	-0.19	36.76	-0.02	-0.27	-0.22	34.96	-0.04	-0.32	-0.36
flower	**G. en V.**	1	1.59	2.4	2.55	1	1.73	2.37	2.7	1	1.54	2.24	2.46	1	1.6	2.15	2.52
250 frames	**PSNR (dB)**	30.7	-0.09	0.01	-0.08	27.7	-0.04	-0.14	-0.15	26.09	-0.04	-0.15	-0.26	24.1	-0.01	-0.31	-0.47
foreman	**G. en V.**	1	1.69	2.53	2.89	1	1.58	2.42	2.59	1	1.52	2.3	2.63	1	1.42	2.06	2.3
300 frames	**PSNR (dB)**	38.55	-0.14	0.06	0.02	35.66	-0.1	-0.12	-0.21	33.92	-0.07	-0.23	-0.36	31.48	0	-0.37	-0.55
mobile	**G. en V.**	1	1.74	2.61	2.77	1	1.71	2.55	2.85	1	1.59	2.37	2.61	1	1.48	2.21	2.45
300 frames	**PSNR (dB)**	30.81	-0.08	-0.2	-0.24	27.71	-0.05	-0.15	-0.24	26.05	-0.04	-0.18	-0.29	23.88	0.09	-0.22	-0.39
news	**G. en V.**	1	1.59	2.07	2.17	1	1.48	1.91	2.14	1	1.44	1.93	2.07	1	1.36	1.85	2.01
300 frames	**PSNR (dB)**	43.05	0.12	-0.02	-0.04	39.89	-0.01	0	-0.13	38.02	0.03	-0.04	-0.16	35.33	0.12	-0.16	-0.35
silent	**G. en V.**	1	1.52	2.17	2.33	1	1.44	2.05	2.31	1	1.46	2.06	2.31	1	1.3	1.9	2.09
300 frames	**PSNR (dB)**	41.23	-0.05	-0.05	-0.31	37.1	-0.01	-0.05	-0.18	35.37	0.04	-0.1	-0.12	33.06	0.02	0.02	-0.17
stefan	**G. en V.**	1	1.67	2.56	2.82	1	1.72	2.47	2.52	1	1.55	2.13	2.53	1	1.46	2.08	2.31
90 frames	**PSNR (dB)**	34.12	-0.1	-0.08	-0.11	30.85	-0.09	-0.22	-0.28	28.96	-0.08	-0.33	-0.42	26.08	0.03	-0.4	-0.57
tempete	**G. en V.**	1	1.72	2.59	2.88	1	1.65	2.46	2.74	1	1.56	2.31	2.64	1	1.46	2.13	2.37
260 frames	**PSNR (dB)**	33.26	-0.07	0	-0.05	30.39	-0.02	-0.11	-0.21	28.79	0	-0.17	-0.32	26.77	0.03	-0.35	-0.63
waterfall	**G. en V.**	1	1.88	2.77	3	1	1.82	2.73	3.02	1	1.68	2.7	2.94	1	1.58	2.37	2.62
260 frames	**PSNR (dB)**	38.8	-0.07	-0.18	-0.2	35.78	-0.05	-0.16	-0.25	33.95	0.06	-0.22	-0.3	31.77	0.13	-0.24	-0.41
Moyenne (G. en V.)		1	1.66	2.39	2.6	1	1.62	2.3	2.52	1	1.53	2.2	2.44	1	1.43	2.04	2.26
Moyenne (PSNR)		36.91	-0.05	-0.06	-0.12	33.71	-0.04	-0.12	-0.2	31.99	-0.01	-0.18	-0.27	29.71	0.04	-0.26	-0.43

G. en V. : Gains en vitesse.

Tableau 6.2 Résultats de l'adaptation du débit binaire utilisant des séquences CIF@256 kb/s et transcodées à 192, 128, 96 et 64 kb/s

Vidéos CIF		256 à 192 kb/s				256 à 128 kb/s				256 à 96 kb/s				256 à 64 kb/s			
		Cas-cade	Stati-stics	Prop-osée	Raff. Cond.	Cas-cade	Stati-stics	Prop-osée	Raff. Cond.	Cas-cade	Stati-stics	Prop-osée	Raff. Cond.	Cas-cade	Stati-stics	Prop-osée	Raff. Cond.
container	**G. en V.**	1	1.43	1.75	1.79	1	1.47	1.8	1.76	1	1.29	1.57	1.63	1	1.28	1.52	1.4
300 frames	**PSNR (dB)**	39.83	0.15	-0.01	-0.01	36.57	-0.04	-0.23	-0.24	34.63	-0.16	-0.31	-0.35	32.83	-0.12	-0.39	-0.34
flower	**G. en V.**	1	1.7	2.31	2.65	1	1.65	2.41	2.68	1	1.55	2.14	2.38	1	1.37	1.9	2.05
250 frames	**PSNR (dB)**	28.54	-0.1	-0.25	-0.3	25.56	-0.09	-0.33	-0.51	23.86	0.01	-0.48	-0.72	21.72	0.12	-0.63	-1.14
foreman	**G. en V.**	1	1.57	2.31	2.52	1	1.53	2.15	2.45	1	1.39	1.93	2.26	1	1.31	1.63	1.92
300 frames	**PSNR (dB)**	35.76	-0.19	-0.04	-0.11	32.56	-0.08	-0.29	-0.48	30.5	-0.04	-0.39	-0.74	27.35	0.03	-0.64	-1.01
mobile	**G. en V.**	1	1.63	2.34	2.5	1	1.61	2.26	2.48	1	1.5	2.05	2.29	1	1.28	1.77	1.91
300 frames	**PSNR (dB)**	28.33	0.05	-0.26	-0.32	25.14	-0.02	-0.29	-0.49	23.23	0.07	-0.26	-0.42	20.75	0.33	-0.34	-0.86
news	**G. en V.**	1	1.45	1.9	2.16	1	1.44	1.85	2.03	1	1.34	1.77	1.88	1	1.25	1.52	1.73
300 frames	**PSNR (dB)**	39.68	0.21	-0.02	-0.04	36.25	-0.11	-0.26	-0.45	34.2	0.13	-0.26	-0.4	31.28	0.11	-0.27	-0.48
silent	**G. en V.**	1	1.5	2.07	2.34	1	1.44	1.94	2.18	1	1.35	1.84	2.03	1	1.24	1.65	1.79
300 frames	**PSNR (dB)**	38.21	-0.01	0.19	-0.08	34.36	0.02	-0.21	-0.29	32.56	-0.01	-0.27	-0.31	30.47	-0.08	-0.32	-0.55
stefan	**G. en V.**	1	1.63	2.1	2.45	1	1.42	2.04	2.13	1	1.38	1.88	1.92	1	1.15	1.62	1.68
90 frames	**PSNR (dB)**	30.6	-0.13	-0.2	-0.25	27.09	0.02	-0.3	-0.53	24.92	0.15	-0.31	-0.72	22.05	0.25	-0.34	-0.69
tempete	**G. en V.**	1	1.65	2.44	2.67	1	1.56	2.25	2.45	1	1.43	2	2.17	1	1.31	1.68	1.76
260 frames	**PSNR (dB)**	30.86	-0.06	-0.16	-0.31	27.95	-0.01	-0.32	-0.56	26.26	0.04	-0.51	-0.91	24.1	0.09	-0.84	-1.38
waterfall	**G. en V.**	1	1.71	2.5	2.74	1	1.73	2.55	2.75	1	1.51	2.19	2.43	1	1.27	1.8	1.86
260 frames	**PSNR (dB)**	35.81	0.01	-0.35	-0.39	32.73	0.14	-0.26	-0.39	30.89	0.24	-0.29	-0.5	28.57	0.63	-0.26	-0.86
Moyenne (G. en V.)		1	1.58	2.19	2.42	1	1.53	2.13	2.32	1	1.41	1.93	2.11	1	1.27	1.67	1.78
Moyenne (PSNR)		34.18	0	-0.12	-0.2	30.91	-0.01	-0.27	-0.43	29	0.04	-0.34	-0.56	26.56	0.15	-0.44	-0.81

G. en V. : Gains en vitesse.

Tableau 6.3 Résultats de l'adaptation du débit binaire utilisant des séquences QCIF@256 kb/s et transcodées à 192, 128, 96 et 64 kb/s

Vidéos QCIF		256 à 192 kb/s				256 à 128 kb/s				256 à 96 kb/s				256 à 64 kb/s			
		Cas-cade	Stati-stics	Prop-osée	Raff. Cond.	Cas-cade	Stati-stics	Prop-osée	Raff. Cond.	Cas-cade	Stati-stics	Prop-osée	Raff. Cond.	Cas-cade	Stati-stics	Prop-osée	Raff. Cond.
container	G. en V.	1	1.62	1.84	1.79	1	1.66	1.89	1.91	1	1.45	1.75	1.72	1	1.35	1.52	1.57
300 frames	PSNR (dB)	43.92	-0.03	-0.19	-0.2	40.99	-0.05	-0.1	-0.1	39.54	0.04	-0.16	-0.17	37.86	-0.07	-0.17	-0.24
foreman	G. en V.	1	1.75	2.47	3.06	1	1.56	2.44	2.72	1	1.52	2.41	2.91	1	1.41	2.14	2.48
300 frames	PSNR (dB)	39.82	-0.21	0.07	0.07	36.8	-0.16	-0.09	-0.16	34.93	-0.11	-0.18	-0.3	32.49	-0.07	-0.21	-0.46
mobile	G. en V.	1	1.85	2.71	2.86	1	1.73	2.53	2.77	1	1.63	2.44	2.82	1	1.57	2.55	2.66
300 frames	PSNR (dB)	32.43	-0.06	-0.13	-0.13	29.51	-0.07	-0.16	-0.18	27.92	-0.08	-0.19	-0.22	26.01	-0.08	-0.25	-0.4
news	G. en V.	1	1.4	2	2.05	1	1.63	2.02	2.28	1	1.47	1.82	1.92	1	1.33	1.61	2
300 frames	PSNR (dB)	43.73	-0.13	-0.15	-0.12	40.39	-0.04	-0.15	-0.11	38.58	-0.16	-0.28	-0.3	35.98	-0.18	-0.21	-0.27
silent	G. en V.	1	1.57	2.11	2.28	1	1.58	2.25	2.62	1	1.41	2.03	2.31	1	1.39	1.98	2.07
300 frames	PSNR (dB)	42.78	-0.07	-0.21	-0.08	38.86	-0.14	-0.16	-0.19	37.13	-0.16	-0.22	-0.27	34.7	-0.14	-0.22	-0.33
Moyenne (G. en V.)		1	1.63	2.22	2.4	1	1.63	2.22	2.46	1	1.49	2.09	2.33	1	1.41	1.96	2.15
Moyenne (PSNR)		40.53	-0.1	-0.12	-0.09	37.31	-0.09	-0.13	-0.14	35.62	-0.09	-0.2	-0.25	33.4	-0.1	-0.21	-0.34

G. en V. : Gains en vitesse.

Tableau 6.4 Résultats de l'adaptation du débit binaire utilisant des séquences QCIF @128 kb/s et transcodées à 96, 64, 48 et 32 kb/s

Vidéos QCIF		128 à 96 kb/s				128 à 64 kb/s				128 à 48 kb/s				128 à 32 kb/s			
		Cas-cade	Stati-stics	Prop-osée	Raff. Cond.	Cas-cade	Stati-stics	Prop-osée	Raff. Cond.	Cas-cade	Stati-stics	Prop-osée	Raff. Cond.	Cas-cade	Stati-stics	Prop-osée	Raff. Cond.
container	G. en V.	1	1.31	1.66	1.54	1	1.46	1.69	1.55	1	1.36	1.53	1.35	1	1.12	1.38	1.51
300 frames	PSNR (dB)	42.57	-0.08	-0.14	-0.15	39.02	-0.05	-0.14	-0.14	37.18	0.07	-0.11	-0.17	35.34	-0.17	-0.31	-0.3
foreman	G. en V.	1	1.6	2.39	2.79	1	1.58	2.34	2.53	1	1.47	2.2	2.55	1	1.38	2	2.25
300 frames	PSNR (dB)	36.63	-0.17	0.05	0.01	33.46	-0.11	-0.21	-0.32	31.51	-0.14	-0.33	-0.54	28.65	0.04	-0.35	-0.75
mobile	G. en V.	1	1.68	2.27	2.64	1	1.63	2.41	2.48	1	1.53	2.27	2.3	1	1.47	1.87	2.36
300 frames	PSNR (dB)	30.19	0	-0.19	-0.27	27.27	-0.14	-0.39	-0.49	25.6	-0.08	-0.41	-0.63	23.34	0.11	-0.42	-0.9
news	G. en V.	1	1.49	1.71	2.08	1	1.65	1.82	2.25	1	1.46	2	2.12	1	1.25	1.63	1.71
300 frames	PSNR (dB)	40.42	-0.12	-0.06	-0.06	36.73	-0.14	-0.22	-0.21	34.84	-0.07	-0.2	-0.32	32.1	-0.04	-0.31	-0.43
silent	G. en V.	1	1.52	2.11	2.39	1	1.39	1.97	2.1	1	1.41	1.79	2.21	1	1.27	1.73	1.93
300 frames	PSNR (dB)	39.37	-0.2	-0.24	-0.11	35.44	-0.02	-0.11	-0.16	33.75	-0.03	-0.2	-0.29	31.19	0	-0.17	-0.25
Moyenne (G. en V.)		1	1.52	2.02	2.28	1	1.54	2.04	2.18	1	1.44	1.95	2.1	1	1.29	1.72	1.95
Moyenne (PSNR)		37.83	-0.11	-0.11	-0.11	34.38	-0.09	-0.21	-0.26	32.57	-0.05	-0.25	-0.39	30.12	-0.01	-0.31	-0.52

G. en V. : Gains en vitesse.

6.2.2 Adaptation de la résolution spatiale

Quatre méthodes ont été implémentées pour la réduction de la résolution spatiale:

- La méthode des statistiques [35] : qui consiste à réutiliser les modes de MBs seulement
- La méthode proposée : qui consiste à réutiliser toutes les métadonnées telle que décrite par la section 4.4.2. Le meilleur VM est raffiné en utilisant une fenêtre de recherche de l'ordre de ±1.
- La méthode proposée avec raffinement conditionnel des VMs : cette méthode est dérivée de la méthode proposée où le VM testé est raffiné seulement s'il est différent du VM prédit. Dans le cas contraire, la précision du VM entrant est réemployée.
- La méthode proposée sans raffinement des VMs : cette méthode est aussi dérivée de la méthode dérivée où aucun raffinement des VMs n'est effectué. Dans ce cas, la précision des VMs entrants est réutilisée.

Les séquences CIF ont été transcodées à des séquences QCIF à des débits binaires variables. Les tableaux 6.5 et 6.6 illustrent les résultats obtenus pour la réduction de la résolution spatiale en H.264. La méthode proposée est 1.4 à 1.6 fois plus rapide que la méthode en cascade avec une dégradation négligeable de la qualité de l'ordre de -0.25 dB à -0.32 dB. La méthode de raffinement conditionnel surpasse la méthode proposée en gain en vitesse qui varie entre 1.66 et 1.9 avec approximativement -0.3 dB de perte en qualité par rapport à cette dernière. Pour la troisième méthode proposée, c.-à-d. sans raffinement des VMs, les gains en vitesse sont les plus élevés mais avec une détérioration autour de -1 dB. On constate donc que le raffinement a un impact très positif sur la qualité de la vidéo mais affecte la vitesse de l'encodeur. De ce fait, dépendamment de nos besoins en vitesse, nous pourrions choisir la première, la deuxième ou la troisième méthode pour réaliser le transcodage.

Tableau 6.5 Résultats de l'adaptation de la résolution spatiale utilisant des séquences CIF@512 kb/s et transcodées à 256, 192 et 128 kb/s

Vidéos CIF		512 à 256 kb/s					512 à 192 kb/s					512 à 128 kb/s				
		Cas-cade	Stati-stics	Prop-osée	Raff. Cond.	Sans Raff.	Cas-cade	Stati-stics	Prop-osée	Raff. Cond.	Sans Raff.	Cas-cade	Stati-stics	Prop-osée	Raff. Cond.	Sans Raff.
container	**G. en V.**	1	1.18	1.29	1.43	1.56	1	1.11	1.11	1.29	1.58	1	1.07	1.19	1.26	1.45
300 frames	**PSNR** (dB)	40.41	-0.14	-0.19	-0.35	-0.66	38.48	-0.08	-0.12	-0.23	-0.65	36.28	-0.05	-0.19	-0.29	-0.74
flower	**G. en V.**	1	1.19	1.54	1.93	2.24	1	1.2	1.62	1.89	2.15	1	1.11	1.58	1.82	1.92
250 frames	**PSNR** (dB)	26.79	-0.15	-0.2	-0.44	-0.79	25.41	-0.13	-0.21	-0.45	-0.86	23.74	-0.03	-0.19	-0.48	-0.9
foreman	**G. en V.**	1	1.32	1.81	2.33	2.74	1	1.24	1.72	2.18	2.41	1	1.22	1.69	2.03	2.11
300 frames	**PSNR** (dB)	35.17	-0.3	-0.64	-1	-1.23	33.8	-0.31	-0.62	-1.01	-1.24	32.02	-0.22	-0.55	-0.94	-1.16
mobile	**G. en V.**	1	1.17	1.68	2.1	2.28	1	1.13	1.67	2.03	2.26	1	1.12	1.56	1.86	2.13
300 frames	**PSNR** (dB)	24.2	-0.07	-0.15	-0.44	-1.06	23.2	-0.05	-0.15	-0.44	-1.09	22.02	-0.02	-0.15	-0.52	-1.23
news	**G. en V.**	1	1.21	1.47	1.64	1.72	1	1.28	1.51	1.62	1.93	1	1.13	1.36	1.51	1.67
300 frames	**PSNR** (dB)	42.8	-0.41	-0.68	-0.9	-1.12	40.29	-0.25	-0.46	-0.78	-0.99	37.5	-0.42	-0.65	-0.93	-1.17
silent	**G. en V.**	1	1.36	1.71	1.91	2.22	1	1.35	1.66	1.89	2.15	1	1.21	1.59	1.78	1.89
300 frames	**PSNR** (dB)	40.64	-0.53	-0.61	-0.85	-1.04	38.35	-0.24	-0.39	-0.7	-0.85	35.89	-0.35	-0.41	-0.73	-0.82
stefan	**G. en V.**	1	1.28	1.66	2.06	2.11	1	1.21	1.58	1.6	2.18	1	1.12	1.35	1.74	1.81
90 frames	**PSNR** (dB)	28.96	-0.12	-0.26	-0.55	-0.93	27.41	-0.12	-0.25	-0.53	-0.99	25.51	0	-0.19	-0.54	-0.95
tempete	**G. en V.**	1	1.2	1.65	1.98	2.28	1	1.2	1.64	1.9	2.22	1	1.12	1.54	1.81	1.92
260 frames	**PSNR** (dB)	29.12	-0.09	-0.1	-0.31	-0.71	27.91	-0.06	-0.07	-0.31	-0.78	26.47	-0.03	-0.05	-0.29	-0.82
waterfall	**G. en V.**	1	1.17	1.59	1.81	2.19	1	1.2	1.59	1.86	2.25	1	1.11	1.48	1.77	1.89
260 frames	**PSNR** (dB)	33.42	-0.06	-0.07	-0.34	-1.05	32.35	-0.04	-0.06	-0.35	-1.2	31.06	-0.02	-0.08	-0.43	-1.32
Moyenne (G. en V.)		1	1.23	1.6	1.91	2.14	1	1.21	1.56	1.8	2.12	1	1.13	1.48	1.73	1.86
Moyenne (PSNR)		33.5	-0.2	-0.32	-0.57	-0.95	31.91	-0.14	-0.25	-0.53	-0.96	30.05	-0.12	-0.27	-0.57	-1.01

G. en V. : Gains en vitesse.

Tableau 6.6 Résultats de l'adaptation de la résolution spatiale utilisant des séquences CIF@256 kb/s et transcodées à 128, 96 et 64 kb/s

Vidéos CIF		256 à 128 kb/s					256 à 96 kb/s					256 à 64 kb/s				
		Cas-cade	Stati-stics	Prop-osée	Raff. Cond.	Sans Raff.	Cas-cade	Stati-stics	Prop-osée	Raff. Cond.	Sans Raff.	Cas-cade	Stati-stics	Prop-osée	Raff. Cond.	Sans Raff.
container	**G. en V.**	1	1.14	1.25	1.32	1.65	1	1.2	1.25	1.28	1.54	1	1.18	1.35	1.39	1.57
300 frames	**PSNR (dB)**	37.47	-0.08	-0.27	-0.44	-0.8	35.86	-0.13	-0.13	-0.36	-0.76	33.84	-0.14	-0.2	-0.41	-0.71
flower	**G. en V.**	1	1.19	1.52	1.83	2.18	1	1.1	1.46	1.73	1.96	1	1.07	1.38	1.59	1.89
250 frames	**PSNR (dB)**	24.85	-0.13	-0.24	-0.6	-1	23.85	-0.12	-0.26	-0.61	-1.08	22.57	-0.09	-0.32	-0.77	-1.22
foreman	**G. en V.**	1	1.21	1.8	2.22	2.32	1	1.23	1.72	2.09	2.15	1	1.18	1.61	1.83	1.99
300 frames	**PSNR (dB)**	32.98	-0.27	-0.56	-0.97	-1.21	31.69	-0.22	-0.49	-0.91	-1.17	29.93	-0.16	-0.46	-0.93	-1.16
mobile	**G. en V.**	1	1.09	1.58	1.79	2.2	1	1.09	1.54	1.86	2.03	1	1.08	1.46	1.67	1.75
300 frames	**PSNR (dB)**	22.81	-0.04	-0.14	-0.51	-1.22	22.07	-0.07	-0.16	-0.58	-1.27	21.09	-0.03	-0.16	-0.63	-1.36
news	**G. en V.**	1	1.18	1.39	1.65	1.81	1	1.17	1.43	1.61	1.71	1	1.08	1.39	1.54	1.57
300 frames	**PSNR (dB)**	38.07	-0.38	-0.65	-0.91	-1.19	36.03	-0.26	-0.42	-0.76	-1.03	33.62	-0.28	-0.54	-0.76	-1.11
silent	**G. en V.**	1	1.21	1.55	1.79	2.14	1	1.24	1.53	1.74	2.01	1	1.19	1.46	1.69	1.88
300 frames	**PSNR (dB)**	36.73	-0.32	-0.41	-0.71	-0.87	35.12	-0.27	-0.42	-0.6	-0.71	32.98	-0.18	-0.32	-0.58	-0.82
stefan	**G. en V.**	1	1.06	1.6	1.89	1.94	1	1.17	1.46	1.68	1.9	1	1.29	1.59	1.81	1.95
90 frames	**PSNR (dB)**	26.34	-0.2	-0.34	-0.7	-1.15	25.21	-0.13	-0.28	-0.66	-1.09	23.74	-0.13	-0.28	-0.65	-1.08
tempete	**G. en V.**	1	1.1	1.53	1.81	2.1	1	1.05	1.37	1.67	1.96	1	1.1	1.49	1.76	1.85
260 frames	**PSNR (dB)**	27.45	-0.02	-0.04	-0.32	-0.79	26.51	-0.03	-0.05	-0.35	-0.86	25.31	-0.01	-0.07	-0.44	-0.98
waterfall	**G. en V.**	1	1.09	1.5	1.77	2.09	1	1.1	1.56	1.65	2.03	1	1.08	1.47	1.69	1.88
260 frames	**PSNR (dB)**	31.8	-0.06	-0.12	-0.44	-1.2	30.95	-0.04	-0.12	-0.51	-1.31	29.87	-0.03	-0.11	-0.6	-1.47
Moyenne (G. en V.)		1	1.14	1.52	1.78	2.04	1	1.15	1.48	1.7	1.92	1	1.13	1.46	1.66	1.81
Moyenne (PSNR)		30.94	-0.16	-0.3	-0.62	-1.04	29.69	-0.14	-0.25	-0.59	-1.03	28.1	-0.11	-0.27	-0.64	-1.1

G. en V. : Gains en vitesse.

6.2.3 Adaptation du format

Les tableaux 6.7 à 6.10 représentent les résultats expérimentaux de l'adaptation du format de H.264 *baseline profile* à MPEG-4 *visual simple profile*. Trois méthodes ont été utilisées, la méthode des statistiques, la méthode proposée et la méthode proposée avec raffinement conditionnel des VMs.

Dans la méthode des statistiques, seule la décision de mode est transmise au ré-encodeur. Un raffinement de ±1 est utilisé pour cette méthode.

Telle que décrite dans la section 4.4.3, la méthode proposée réemploie les modes de MBs, VMs et le reste des métadonnées. Les VMs choisis sont raffinés en utilisant une petite fenêtre de recherche de ±1.

La méthode proposée avec raffinement conditionnel est dérivée de la méthode proposée où le VMs choisis sont raffinés si et seulement si le présent VM évalué diffère du VM prédit.

Les résultats obtenus, soit le gain en vitesse de transcodage et la qualité en décibels, sont comparés pour chaque méthode par rapport à la méthode en cascade où une ré-estimation de mouvement est effectuée. Les séquences CIF ont été encodées à 512 et 256 Kb/s respectivement avec une trame intra chaque 100 trames et étaient transcodées à différents débits binaires. Les tableaux 6.7 et 6.8 illustrent les résultats obtenus. De même, nous avons encodé les séquences QCIF à 256 et 128 kb/s respectivement et nous avons transcodé ces séquences à des débits binaires allant de 256 jusqu'à 32 kb/s. Les résultats obtenus sont illustrés par les tableaux 6.9 et 6.10.

En analysant les quatre tableaux, on peut constater que la méthode proposée est 1.26 à 1.8 fois plus rapide que la méthode cascadée. La raison pour laquelle les gains sans moins impressionnants que les résultats obtenus pour H.264 est le fait que MPEG-4 a moins de modes de MBs à tester (inter16x16 et inter8x8) par rapport à H.264 qui en a dix modes. Ainsi, du fait qu'Intel® IPP est très optimisé, les gains obtenus étaient moins comparativement à H.264. En ce qui concerne la qualité du transcodage, la dégradation varie entre -1.14 dB dans lorsqu'on transcode à des hauts débits et -0.56 dB lorsqu'on transcode à des débits faibles. La méthode du raffinement conditionnel est plus rapide que la méthode proposée mais la dégradation de la qualité est plus élevée. La dégradation de qualité mérite

que l'on améliore encore les méthodes en analysant le comportement de l'encodeur MPEG-4 d'Intel® IPP. Ceci devrait améliorer nettement les résultats obtenus en qualité pour atteindre des pertes négligeables comme celles obtenues pour l'adaptation du débit binaire, et aussi les gains en vitesse.

Ainsi, un compromis devrait être fait entre une grande vitesse et une perte de qualité faible.

Tableau 6.7 Résultats de l'adaptation de format H.264 à MPEG-4 utilisant des séquences CIF@512 kb/s et transcodées à 512, 256, 192 et 128 kb/s

Vidéos CIF		512 à 512 kb/s				512 à 256 kb/s				512 à 192 kb/s				512 à 128 kb/s			
		Cas-cade	Stati-stics	Prop-osée	Raff. Cond.	Cas-cade	Stati-stics	Prop-osée	Raff. Cond.	Cas-cade	Stati-stics	Prop-osée	Raff. Cond.	Cas-cade	Stati-stics	Prop-osée	Raff. Cond.
container	**G. en V.**	1	1.52	1.96	1.95	1	1.37	1.74	1.91	1	1.37	1.68	1.84	1	1.24	1.52	1.69
300 frames	**PSNR (dB)**	39.49	0.02	-0.23	-1.28	35.81	-0.04	-0.22	-1.89	34.51	-0.05	-0.35	-2.16	32.82	-0.1	-0.44	-2.53
flower	**G. en V.**	1	1.25	1.47	1.78	1	1.36	1.74	1.79	1	1.28	1.65	1.78	1	1.45	1.83	2.06
250 frames	**PSNR (dB)**	28.95	-0.24	-1.49	-2.06	25.92	-0.12	-1.09	-1.59	24.91	-0.07	-0.67	-1	24.21	-0.02	-0.19	-0.45
foreman	**G. en V.**	1	1.34	1.84	1.97	1	1.2	1.48	1.76	1	1.09	1.56	1.55	1	1	1.34	1.46
300 frames	**PSNR (dB)**	37.17	-0.26	-1.79	-2.22	33.77	-0.19	-1.77	-2.29	32.39	-0.19	-1.67	-2.25	30.31	-0.24	-1.29	-1.79
mobile	**G. en V.**	1	1.53	2.08	2.1	1	1.34	1.84	1.94	1	1.37	1.82	1.9	1	1.33	1.86	1.75
300 frames	**PSNR (dB)**	26.99	-0.1	-0.91	-1.29	24.34	-0.06	-0.58	-0.84	23.71	-0.02	-0.22	-0.4	23.55	-0.01	-0.13	-0.31
news	**G. en V.**	1	1.33	1.65	1.78	1	1.25	1.64	1.67	1	1.2	1.65	1.83	1	1.12	1.45	1.64
300 frames	**PSNR (dB)**	42.07	-0.18	-1.66	-1.83	37.57	-0.12	-1.78	-2.03	35.91	-0.17	-1.66	-2.04	33.62	-0.19	-1.53	-2.03
silent	**G. en V.**	1	1.52	2.11	2.25	1	1.38	1.97	2.12	1	1.23	1.83	1.99	1	1.12	1.58	1.67
300 frames	**PSNR (dB)**	39.89	-0.19	-1.35	-1.24	35.59	-0.08	-1.16	-1.01	34.08	-0.12	-1.14	-1.07	32.14	-0.15	-0.92	-1.1
stefan	**G. en V.**	1	1.28	1.5	1.57	1	1.32	1.55	1.63	1	1.23	1.48	1.52	1	1.34	1.6	1.66
90 frames	**PSNR (dB)**	32.32	-0.18	-1.53	-1.91	28.56	-0.12	-1.34	-1.88	27.21	-0.14	-1.23	-1.72	25.47	-0.02	-0.25	-0.52
tempete	**G. en V.**	1	1.48	1.87	1.99	1	1.32	1.64	1.71	1	1.25	1.54	1.64	1	1.1	1.42	1.42
260 frames	**PSNR (dB)**	31.36	-0.08	-0.82	-1.2	28.46	-0.03	-0.61	-1.07	27.4	-0.02	-0.5	-0.97	26.05	-0.03	-0.29	-0.5
waterfall	**G. en V.**	1	1.63	2.08	2.19	1	1.4	1.77	1.87	1	1.25	1.61	1.68	1	1.15	1.47	1.53
260 frames	**PSNR (dB)**	35.21	-0.11	-0.48	-1.69	32.36	-0.11	-0.43	-1.65	31.27	-0.08	-0.38	-1.56	29.76	-0.04	-0.31	-1.46
Moyenne (G. en V.)		1	1.43	1.84	1.95	1	1.32	1.7	1.82	1	1.25	1.64	1.74	1	1.2	1.56	1.65
Moyenne (PSNR)		34.82	-0.14	-1.14	-1.63	31.37	-0.09	-0.99	-1.58	30.15	-0.09	-0.86	-1.46	28.65	-0.08	-0.59	-1.18

G. en V. : Gains en vitesse.

Tableau 6.8 Résultats de l'adaptation de format H.264 à MPEG-4 utilisant des séquences CIF@256 kb/s et transcodées à 256, 128, 96 et 64 kb/s

Vidéos CIF		256 à 256 kb/s				256 à 128 kb/s				256 à 96 kb/s				256 à 64 kb/s			
		Cas-cade	Stati-stics	Prop-osée	Raff. Cond.	Cas-cade	Stati-stics	Prop-osée	Raff. Cond.	Cas-cade	Stati-stics	Prop-osée	Raff. Cond.	Cas-cade	Stati-stics	Prop-osée	Raff. Cond.
container	**G. en V.**	1	1.37	1.64	1.95	1	1.29	1.62	1.65	1	1.25	1.49	1.69	1	1.16	1.62	1.69
300 frames	**PSNR (dB)**	36.87	-0.07	-0.25	-1.97	33.54	-0.05	-0.36	-2.54	32.13	0.02	-0.23	-2.62	30.26	-0.09	-0.33	-2.56
flower	**G. en V.**	1	1.44	1.84	1.89	1	1.44	1.83	2	1	1.33	1.63	2.01	1	1.22	1.5	1.72
250 frames	**PSNR (dB)**	27.29	-0.22	-0.71	-1.62	24.94	-0.02	-0.17	-0.51	24.81	-0.01	-0.09	-0.43	24.77	-0.01	-0.09	-0.42
foreman	**G. en V.**	1	1.12	1.59	1.7	1	1.1	1.52	1.5	1	1.12	1.4	1.47	1	1.04	1.28	1.33
300 frames	**PSNR (dB)**	34.95	-0.25	-1.29	-2.07	31.12	-0.14	-1.12	-1.78	29.45	-0.13	-0.57	-0.91	28.63	-0.04	-0.05	-0.17
mobile	**G. en V.**	1	1.43	1.9	2	1	1.28	1.46	1.72	1	1.2	1.51	1.8	1	1.35	1.54	1.77
300 frames	**PSNR (dB)**	25.35	-0.06	-0.48	-1.05	24.15	0	-0.1	-0.32	24.1	0	-0.09	-0.31	24.07	0	-0.09	-0.3
news	**G. en V.**	1	1.18	1.49	1.75	1	1.15	1.46	1.8	1	1.12	1.65	1.74	1	1.1	1.4	1.47
300 frames	**PSNR (dB)**	38.36	-0.18	-1.67	-1.95	34.12	-0.18	-1.42	-1.98	32.38	-0.11	-1.22	-1.86	30.13	-0.15	-1.02	-1.53
silent	**G. en V.**	1	1.26	1.77	1.91	1	1.11	1.54	1.8	1	1.08	1.45	1.55	1	1	1.32	1.55
300 frames	**PSNR (dB)**	36.92	-0.19	-1.19	-1.3	32.95	-0.09	-0.8	-1.04	31.47	-0.11	-0.66	-1.02	29.33	-0.03	-0.24	-0.58
stefan	**G. en V.**	1	1.3	1.5	1.46	1	1.34	1.6	1.51	1	1.11	1.32	1.44	1	1.26	1.25	1.44
90 frames	**PSNR (dB)**	29.41	-0.2	-0.83	-1.73	26.05	-0.06	-0.28	-0.71	25.72	-0.02	-0.11	-0.48	25.59	0.01	-0.09	-0.44
tempete	**G. en V.**	1	1.27	1.62	1.65	1	1.17	1.47	1.69	1	1.11	1.41	1.6	1	1.13	1.49	1.58
260 frames	**PSNR (dB)**	29.7	-0.07	-0.41	-1.19	26.89	-0.02	-0.32	-0.73	26.28	0	-0.13	-0.28	26.2	0.01	-0.1	-0.23
waterfall	**G. en V.**	1	1.34	1.66	1.83	1	1.12	1.38	1.65	1	0.99	1.42	1.51	1	1.06	1.37	1.43
260 frames	**PSNR (dB)**	33.16	-0.1	-0.28	-1.67	30.36	-0.05	-0.21	-1.54	29.25	-0.03	-0.35	-1.65	27.84	-0.01	-0.76	-0.88
Moyenne (G. en V.)		1	1.3	1.66	1.79	1	1.22	1.54	1.7	1	1.14	1.47	1.64	1	1.14	1.41	1.55
Moyenne (PSNR)		32.44	-0.14	-0.79	-1.61	29.34	-0.06	-0.53	-1.23	28.39	-0.04	-0.38	-1.06	27.42	-0.03	-0.3	-0.79

G. en V. : Gains en vitesse.

Tableau 6.9 Résultats de l'adaptation de format H.264 à MPEG-4 utilisant des séquences QCIF@256 kb/s et transcodées à 256, 128, 96 et 64 kb/s

Vidéos QCIF		256 à 256 kb/s				256 à 128 kb/s				256 à 96 kb/s				256 à 64 kb/s			
		Cas-cade	Stati-stics	Prop-osée	Raff. Cond.	Cas-cade	Stati-stics	Prop-osée	Raff. Cond.	Cas-cade	Stati-stics	Prop-osée	Raff. Cond.	Cas-cade	Stati-stics	Prop-osée	Raff. Cond.
container	**G. en V.**	1	1.38	1.56	1.44	1	1.24	1.32	1.54	1	1.13	1.2	1.45	1	1.08	1.31	1.33
300 frames	**PSNR (dB)**	42.9	-0.31	-0.45	-0.5	38.93	-0.14	-0.45	-0.76	37.34	-0.09	-0.39	-1.05	35.23	-0.12	-0.45	-1.33
foreman	**G. en V.**	1	1.32	1.46	1.56	1	1.24	1.54	1.44	1	1.07	1.36	1.28	1	1.08	1.29	1.17
300 frames	**PSNR (dB)**	37.85	-0.3	-1.98	-2.21	34.42	-0.24	-2.03	-2.42	33	-0.25	-1.96	-2.38	31.03	-0.21	-1.69	-2.2
mobile	**G. en V.**	1	1.36	1.63	1.54	1	1.31	1.61	1.74	1	1.54	1.6	1.76	1	1.34	1.67	1.79
300 frames	**PSNR (dB)**	28.65	-0.08	-0.8	-1.21	25.76	-0.05	-0.66	-1.17	24.76	-0.04	-0.58	-1.09	23.51	-0.04	-0.54	-0.93
news	**G. en V.**	1	1.21	1.39	1.24	1	1.23	1.25	1.3	1	1.09	1.34	1.17	1	1.18	1.34	1.38
300 frames	**PSNR (dB)**	42.59	-0.15	-1.21	-1.3	38.18	-0.16	-1.33	-1.56	36.13	-0.24	-1.24	-1.56	33.75	-0.15	-1.11	-1.6
silent	**G. en V.**	1	1.31	1.43	1.64	1	1.31	1.38	1.45	1	1.39	1.6	1.79	1	1.21	1.38	1.34
300 frames	**PSNR (dB)**	41.95	-0.24	-1.21	-1.15	37.94	-0.09	-1.35	-1.31	35.96	-0.29	-1.19	-1.12	33.56	-0.05	-1.05	-0.97
Moyenne (G. en V.)		1	1.31	1.49	1.48	1	1.26	1.42	1.49	1	1.24	1.42	1.49	1	1.17	1.39	1.4
Moyenne (PSNR)		38.78	-0.21	-1.13	-1.27	35.04	-0.13	-1.16	-1.44	33.43	-0.18	-1.07	-1.44	31.41	-0.11	-0.96	-1.4

G. en V. : Gains en vitesse.

Tableau 6.10 Résultats de l'adaptation de format H.264 à MPEG-4 utilisant des séquences QCIF@128 kb/s et transcodées à 128, 64, 48 et 32 kb/s

Vidéos QCIF		128 à 128 kb/s				128 à 64 kb/s				128 à 48 kb/s				128 à 32 kb/s			
		Cas-cade	Stati-stics	Prop-osée	Raff. Cond.	Cas-cade	Stati-stics	Prop-osée	Raff. Cond.	Cas-cade	Stati-stics	Prop-osée	Raff. Cond.	Cas-cade	Stati-stics	Prop-osée	Raff. Cond.
container	**G. en V.**	1	1.25	1.44	1.58	1	1.16	1.31	1.31	1	1.2	1.53	1.58	1	1.16	1.32	1.34
300 frames	**PSNR (dB)**	40.31	-0.34	-0.57	-1.21	35.92	-0.17	-0.44	-1.52	34.41	-0.23	-0.49	-1.8	32.3	-0.2	-0.6	-2.16
foreman	**G. en V.**	1	1.48	1.64	1.57	1	1.07	1.15	1.2	1	1.16	1.42	1.44	1	1.07	1.06	1.22
300 frames	**PSNR (dB)**	35.28	-0.29	-1.57	-2.18	31.68	-0.2	-1.39	-2.02	30.19	-0.18	-1.22	-1.83	28.02	-0.15	-0.63	-1.07
mobile	**G. en V.**	1	1.26	1.43	1.59	1	1.25	1.47	1.5	1	1.19	1.5	1.5	1	1.2	1.33	1.45
300 frames	**PSNR (dB)**	26.75	-0.08	-0.53	-1.18	24.24	-0.03	-0.38	-0.91	23.43	-0.02	-0.19	-0.52	23.07	-0.01	-0.07	-0.29
news	**G. en V.**	1	1.19	1.44	1.4	1	1.15	1.34	1.43	1	1.18	1.4	1.49	1	1.11	1.3	1.41
300 frames	**PSNR (dB)**	38.82	-0.22	-1.15	-1.39	34.3	-0.28	-1.07	-1.53	32.53	-0.12	-0.91	-1.41	30.31	-0.12	-0.85	-1.4
silent	**G. en V.**	1	1.33	1.63	1.66	1	1.21	1.44	1.51	1	1.18	1.27	1.51	1	1.07	1.29	1.35
300 frames	**PSNR (dB)**	38.56	-0.21	-0.92	-1.28	34.21	-0.22	-0.96	-1.17	32.46	-0.12	-0.85	-0.98	30.3	-0.07	-0.66	-0.85
Moyenne (G. en V.)		1	1.3	1.51	1.56	1	1.16	1.34	1.39	1	1.18	1.42	1.5	1	1.12	1.26	1.35
Moyenne (PSNR)		35.94	-0.22	-0.94	-1.44	32.07	-0.18	-0.84	-1.43	30.6	-0.13	-0.73	-1.3	28.8	-0.11	-0.56	-1.15

G. en V. : Gains en vitesse.

6.2.4 Adaptation du format avec adaptation de la résolution spatiale

Deux méthodes ont été implémentées pour l'adaptation de format H.264 à MPEG-4 avec adaptation de la résolution spatiale:

- La méthode des statistiques : qui consiste à réutiliser les modes de MBs seulement
- La méthode proposée : qui consiste à réutiliser toutes les métadonnées telle que décrite par la section 4.4.4. Le meilleur VM est raffiné en utilisant une fenêtre de recherche de l'ordre de ±1.

Les séquences CIF ont été transcodées à des séquences QCIF à des débits binaires variables. Les tableaux 6.11 et 6.12 illustrent les résultats obtenus pour l'adaptation de format avec adaptation de la résolution spatiale. La méthode proposée est à peu près 1.2 fois plus rapide que la méthode en cascade avec une dégradation moyenne de la qualité de l'ordre de -2 dB en PSNR. Cette dégradation est due particulièrement à l'encodeur MPEG-4. Plusieurs mois d'analyse de la situation n'ont pas permis de complètement comprendre les raisons de cette perte et surtout d'améliorer les résultats obtenus en vitesse et en qualité. Une analyse plus profonde de cet encodeur vis-à-vis des algorithmes proposés serait requise.

Tableau 6.11 Résultats de l'adaptation de format H.264 à MPEG-4 avec adaptation de la résolution spatiale, utilisant des séquences CIF@512 kb/s et transcodées à 256, 128, 96 et 64 kb/s

Vidéos CIF		512 à 256 kb/s			512 à 128 kb/s			512 à 96 kb/s			512 à 64 kb/s		
		Cas-cade	Stati-stics	Prop-osée	Cas-cade	Stati-stics	Prop-osée	Cas-cade	Stati-stics	Prop-osée	Cas-cade	Stati-stics	Prop-osée
container	G. en V.	1	0.88	1.25	1	1.03	1.23	1	1.06	1.09	1	0.97	1.19
300 frames	PSNR (dB)	38.79	-0.04	-1.93	34.96	-0.12	-1.9	33.38	-0.13	-1.59	31.28	-0.01	-1.42
flower	G. en V.	1	1	1.23	1	0.93	1.03	1	1	1.19	1	1.04	1.14
250 frames	PSNR (dB)	25.98	0	-3.02	23.15	-0.01	-1.57	22.34	0	-0.87	21.89	0	0.51
foreman	G. en V.	1	1.01	1.28	1	1.09	1.29	1	1.05	1.45	1	1	1.17
300 frames	PSNR (dB)	33.5	-0.01	-2.52	30.76	0	2.07	29.72	-0.02	-1.77	28.32	-0.01	-1.37
mobile	G. en V.	1	1.05	1.33	1	1	1.28	1	1	1.21	1	1.02	1.33
300 frames	PSNR (dB)	22.63	-0.01	-1.07	20.63	0.01	-0.17	20.54	0	-0.15	20.46	0	0.16
news	G. en V.	1	1.16	1.3	1	0.99	1.21	1	0.98	1.21	1	1	1.26
300 frames	PSNR (dB)	39.5	-0.06	-1.95	35.1	0.01	-1.92	33.29	-0.01	-1.7	31.1	0.01	-1.27
silent	G. en V.	1	1.11	1.25	1	1.05	1.12	1	1.15	1.42	1	1.11	1.24
300 frames	PSNR (dB)	38.67	0	-2.01	34.69	-0.14	-1.65	32.9	0.02	-1.48	30.97	0.01	-1.22
stefan	G. en V.	1	0.97	1.15	1	1.05	1.17	1	0.88	1.02	1	1	1.14
90 frames	PSNR (dB)	28.6	-0.01	-2.2	25.37	-0.01	-1.86	24.24	0	-1.43	22.97	0	0.64
tempete	G. en V.	1	1.02	1.13	1	1.01	1.24	1	0.98	1.2	1	1.02	1.11
260 frames	PSNR (dB)	28.09	-0.01	-1.72	25.66	0	0.99	24.88	-0.01	-0.75	23.94	-0.01	-0.49
waterfall	G. en V.	1	1.02	1.26	1	1.11	1.4	1	1.05	1.3	1	1.05	1.35
260 frames	PSNR (dB)	31.74	-0.02	-2.19	29.58	0	-2	28.84	0	-1.82	27.97	0.01	-1.54
Moyenne (G. en V.)		1	1.02	1.24	1	1.02	1.21	1	1.01	1.23	1	1.02	1.21
Moyenne (PSNR)		31.94	-0.01	-2.06	28.87	-0.02	-1.57	27.79	-0.01	-1.28	26.54	0	0.95

G. en V. : Gains en vitesse.

Tableau 6.12 Résultats de l'adaptation de format H.264 à MPEG-4 avec adaptation de la résolution spatiale, utilisant des séquences CIF@256 kb/s et transcodées à 128, 64, 48 et 32 kb/s

Vidéos CIF		256 à 128 kb/s			256 à 64 kb/s			256 à 48 kb/s			256 à 32 kb/s		
		Cas-cade	Stati-stics	Prop-osée	Cas-cade	Stati-stics	Prop-osée	Cas-cade	Stati-stics	Prop-osée	Cas-cade	Stati-stics	Prop-osée
container	G. en V.	1	0.88	1.08	1	1.15	1.28	1	1	1.26	1	1.04	1.13
300 frames	PSNR (dB)	35.85	-0.03	-1.79	31.73	-0.03	-1.49	30.35	-0.14	-1.19	28.43	-0.08	-0.83
flower	G. en V.	1	1	1.22	1	1	1.2	1	1.02	1.24	1	0.95	1.12
250 frames	PSNR (dB)	24.05	0	-2.05	22.38	0	-0.61	22.15	0	-0.47	22.05	0	-0.44
foreman	G. en V.	1	0.92	1.14	1	1	1.27	1	1.06	1.19	1	1.03	1.21
300 frames	PSNR (dB)	31.51	0	-2.14	28.88	-0.02	-1.4	27.79	0	-0.96	26.35	0.01	-0.29
mobile	G. en V.	1	1	1.29	1	1.05	1.31	1	1.02	1.26	1	1.01	1.25
300 frames	PSNR (dB)	21.29	-0.01	-0.45	20.83	-0.01	-0.14	20.78	0	-0.14	20.72	-0.01	-0.15
news	G. en V.	1	0.94	1.18	1	1.01	1.23	1	1.09	1.15	1	1	1.09
300 frames	PSNR (dB)	35.48	0.01	-1.68	31.53	-0.04	-1.31	30.11	-0.04	-1.26	28.11	-0.04	-0.73
silent	G. en V.	1	1.01	1.17	1	1.09	1.31	1	1.06	1.3	1	1.1	1.22
300 frames	PSNR (dB)	35.19	0.06	-1.45	31.58	0	-1.07	30.34	0	-0.96	28.58	-0.04	-0.51
stefan	G. en V.	1	0.97	1.05	1	0.9	1.07	1	1.06	1.2	1	1.09	1.19
90 frames	PSNR (dB)	26.03	-0.01	-1.78	23.5	0	-0.84	22.76	0.01	-0.29	22.44	0.01	-0.16
tempete	G. en V.	1	1	1.26	1	1.09	1.22	1	1.08	1.21	1	0.97	1.2
260 frames	PSNR (dB)	26.46	-0.02	-1.21	24.59	-0.01	-0.47	23.95	-0.01	-0.24	23.33	0.02	0.15
waterfall	G. en V.	1	1.07	1.29	1	1.02	1.27	1	0.98	1.29	1	1.05	1.34
260 frames	PSNR (dB)	30.17	-0.01	-2.04	28.45	0	-1.63	27.84	0	-1.27	27	-0.01	-0.79
Moyenne (G. en V.)		1	0.97	1.18	1	1.03	1.24	1	1.04	1.23	1	1.02	1.19
Moyenne (PSNR)		29.55	0	-1.62	27.05	-0.01	-0.99	26.23	-0.02	-0.75	25.22	-0.01	-0.41

G. en V. : Gains en vitesse.

6.3 Conclusion

Dans ce chapitre, nous avons présenté les simulations et résultats obtenus de chaque fonctionnalité ou groupe de fonctionnalités de transcodage vidéo incluant la réduction du débit binaire (RDB) en H.264, l'adaptation de la résolution spatiale avec l'adaptation du débit binaire en H.264, l'adaptation de format de H.264 (*baseline profile*) à MPEG-4 (*visual simple profile*) avec adaptation du débit binaire et l'adaptation de format de H.264 (*baseline profile*) à MPEG-4 (*visual simple profile*) avec adaptation de la résolution spatiale et du débit binaire. En général, l'architecture unifiée avec les différents algorithmes qui ont été implémentés surpasse l'architecture cascadée avec une dégradation faible dans la plupart des cas.

CONCLUSION

Dans ce mémoire, nous avons présenté un nouveau système de transcodage vidéo, appelé système unifié, capable de réaliser une simple opération ou un groupe d'opérations de transcodage vidéo. L'objectif de ce projet de recherche était de minimiser l'effort de développement et de maintenance du logiciel tout en maximisant la rapidité du transcodeur et en minimisant la dégradation de la qualité.

Après une revue de la littérature, nous avons vu, dans le chapitre 2, que plusieurs algorithmes ont été proposés pour réaliser les fonctionnalités de transcodage vidéo. Les algorithmes proposés se sont concentrés sur la réalisation de chaque opération indépendamment des autres, malgré qu'il y ait un grand nombre de traitements partagés entre les opérations, ce qui force à créer autant de transcodeurs que de cas d'utilisation. Rares sont les travaux qui ont essayé de proposer une solution globale au problème de transcodage vidéo. Le chapitre 3 a présenté quelques travaux qui ont implémenté plus d'une opération dans le même transcodeur. Cependant, certaines opérations y manquaient, notamment l'insertion de logo et l'adaptation de format en supportant les standards les plus utilisés, particulièrement H.264 et VC-1.

Ainsi, un transcodeur universel capable de réaliser une seule opération ou un ensemble d'opérations de manière fiable et efficace, c.-à-d. en réduisant les calculs faits par le transcodeur tout en maintenant une très haute qualité de la vidéo est indispensable. En plus, notre système unifié se veut un système flexible et capable de supporter les besoins futures sans tout développer de la base. Les résultats obtenus ont montré que notre système unifié est capable d'atteindre jusqu'à 300% de gain de vitesse avec une perte mineure de l'ordre de -0.06 dB pour l'adaptation du débit binaire par exemple.

Les perspectives futures de ce travail consistent à améliorer notre système unifié en développant des algorithmes spécifiques pour le transcodage H.264 à MPEG-4 et pour d'autres cas d'utilisation d'intérêt. Ceci devrait être fait en analysant plus en détail le

fonctionnement de l'encodeur MPEG-4 d'Intel® IPP. En plus, l'extension des fonctionnalités réalisées en suivant les mêmes étapes que ce travail pourrait donner une idée plus précise des gains en vitesse et de la qualité de la vidéo obtenue. Aussi, il serait très intéressant de réaliser un transcodage intelligent en détectant la meilleure combinaison d'opérations possible qui pourrait être faite pour réaliser un ensemble d'opérations demandé. Par exemple, si l'usager final demande une adaptation de format avec réduction spatio-temporelle, on pourrait lui suggérer, après négociation, une adaptation de format avec réduction spatiale et réduction du débit binaire car on sait que la qualité sera meilleure.

ANNEXE I

BLOC DE RÉCUPÉRATION DES MÉTADONNÉES DANS LA PLATEFORME UVECT : CLASSES UTILISÉES

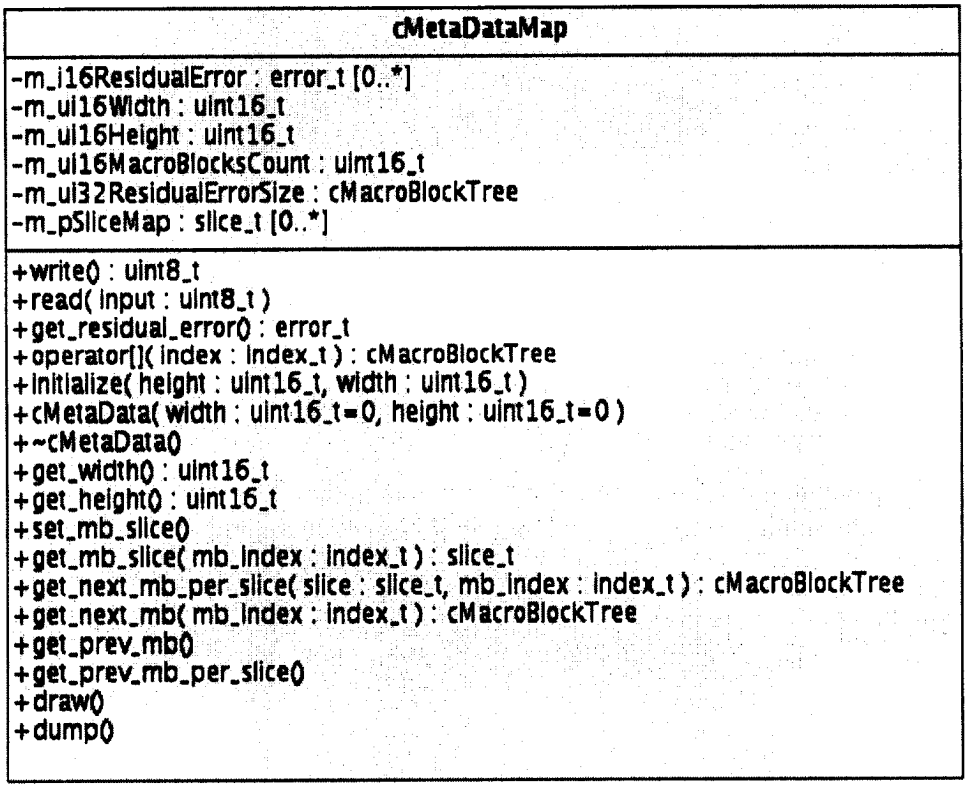

Figure-A I-6.1 Classe MetaDataMap.

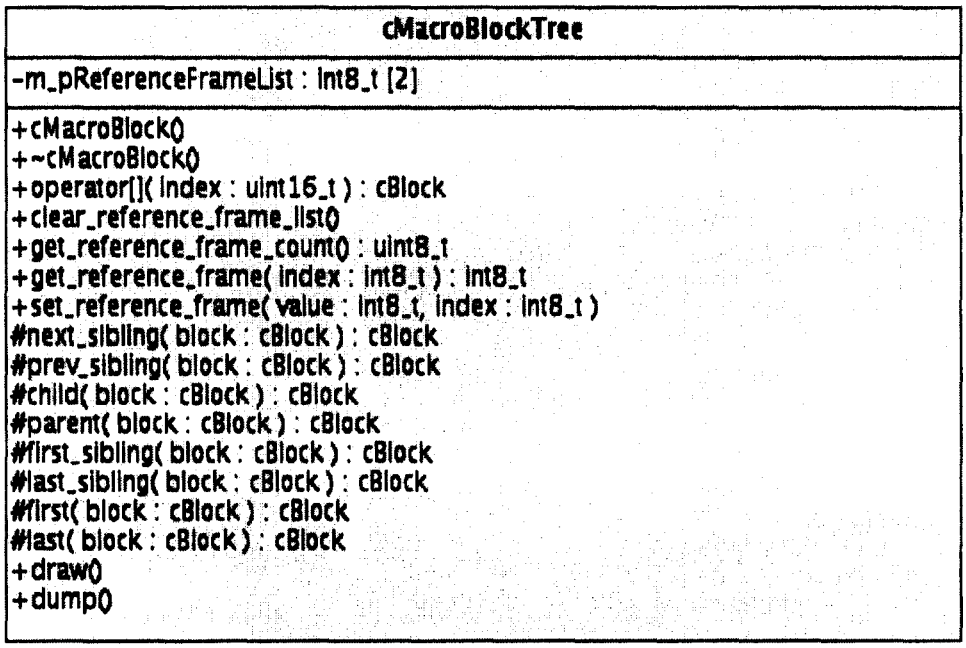

Figure-A I-6.2 Classe MacroBlockTree.

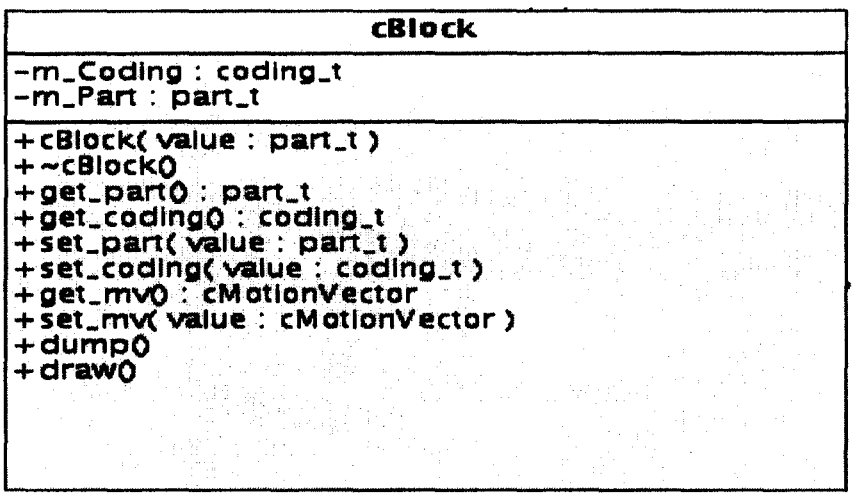

Figure-A I-6.3 Classe Block.

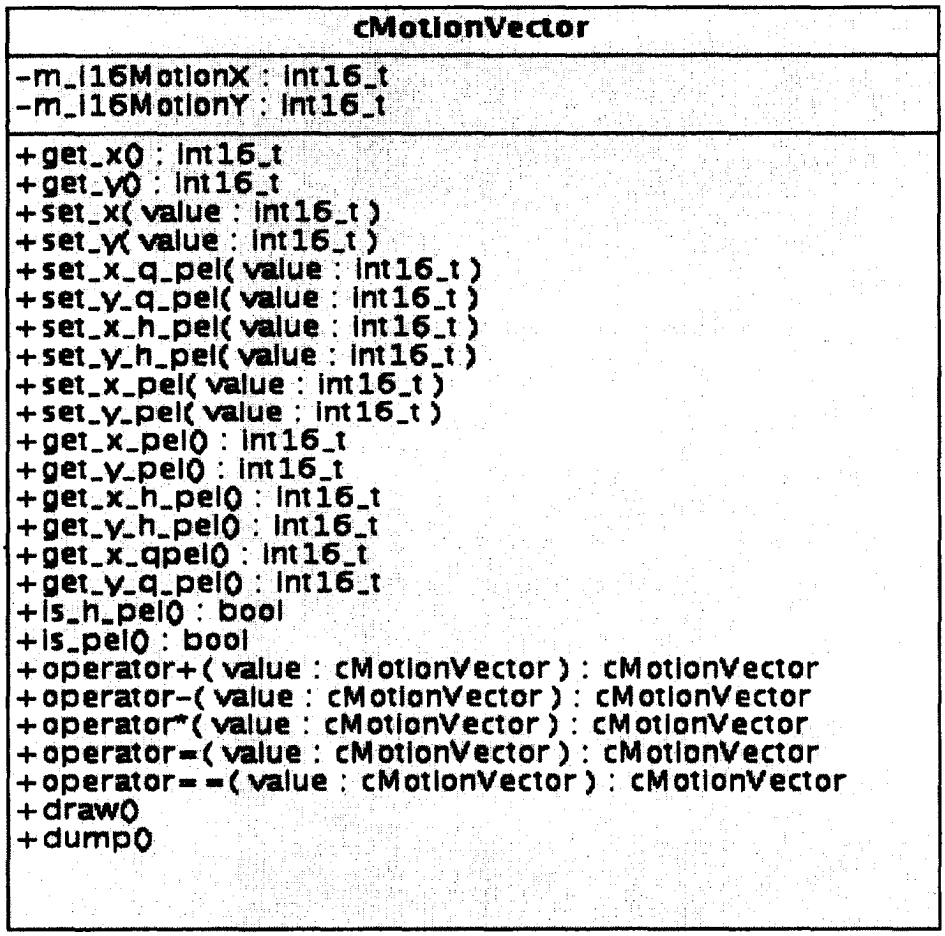

Figure-A I-6.4 Classe MotionVector.

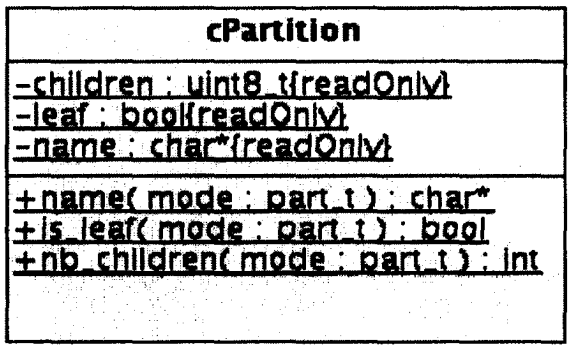

Figure-A I-6.5 Classe Partition.

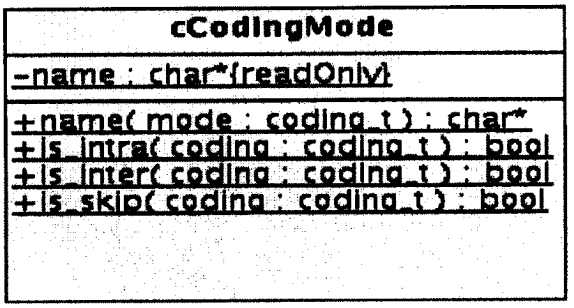

Figure-A I-6.6 Classe CodingMode.

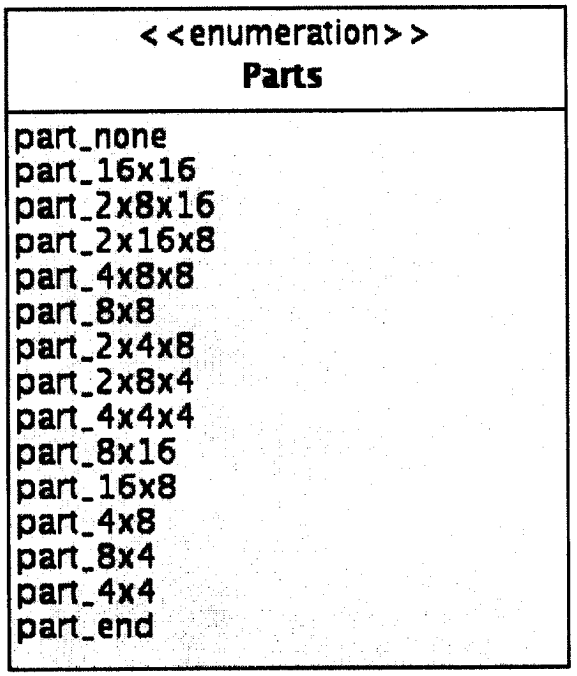

Figure-A I-6.7 Énumération Parts.

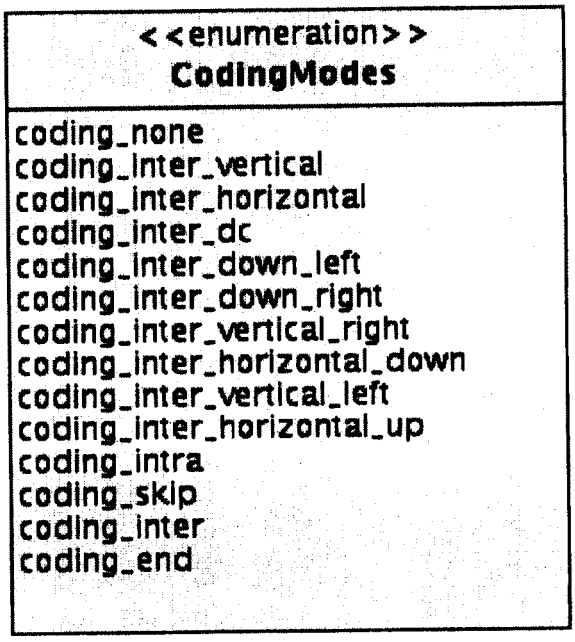

Figure-A I-6.8 Énumération CodingModes.

BIBLIOGRAPHIE

[1] Ahmad, Ishfaq, Xiaohui Wei, Yu Sun et Ya-Qin Zhang. 2005. « Video transcoding: An overview of various techniques and research issues ». *IEEE Transactions on Multimedia,* vol. 7, n° 5, p. 793-804.

[2] Assuncao, Pedro A. A., et Mohammed Ghanbari. 1996. « Post-processing of MPEG2 coded video for transmission at lower bit rates ». In *Proceedings of the 1996 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP. Part 4 (of 6)* (Atlanta, GA, May 7-10 1996). Vol. 4, p. 1998-2001. IEEE, Piscataway, NJ, USA.

[3] Bialkowski, Jens, Marcus Barkowsky et Andre Kaup. 2006. « Overview of low-complexity video transcoding from H.263 to H.264 ». In *2006 IEEE International Conference on Multimedia and Expo, ICME 2006* (Piscataway, NJ 08855-1331, Jul 9-12 2006). Vol. 2006, p. 49-52. Institute of Electrical and Electronics Engineers Computer Society.

[4] Bialkowski, Jens, Marcus Barkowsky, Florian Leschka et André Kaup. 2006. « Low-complexity transcoding of inter coded video frames from H.264 to H.263 ». In *2006 International Conference on Image Processing* (Piscataway, NJ, 8-11 Oct. 2006). p. 837-840. IEEE.

[5] Bialkowski, Jens, Andre Kaup et Klaus Illgner. 2004. « Fast transcoding of intra frames between H.263 and H.264 ». In *2004 International Conference on Image Processing, ICIP 2004, Oct 18-21 2004* (Piscataway, NJ 08855-1331). Vol. 4, p. 2785-2788. Institute of Electrical and Electronics Engineers Computer Society.

[6] Bjork, Niklas, et Charilaos Christopoulos. 1998. « Transcoder architectures for video coding ». *IEEE Transactions on Consumer Electronics,* vol. 44, n° 1, p. 88-98.

[7] Chang, Shih-Fu, et Anthony Vetro. 2005. « Video adaptation: Concepts, technologies, and open issues ». *Proceedings of the IEEE,* vol. 93, n° 1, p. 148-158.

[8] Chih-Hung, Li, Wang Chung-Neng et Chiang Tihao. 2004. « A fast downsizing video transcoder based on H.264/AVC standard ». In *Advances in Multimedia Information Processing - PCM 2004. 5th Pacific Rim Conference on Multimedia. Proceedings, Part III* (Tokyo, Japan, 30 Nov.-3 Dec. 2004). p. 215-23. Springer-Verlag.

[9] Choupani, Roya, Stephan Wong et Mehmet Tolun. 2007. « Video Coding and Transcoding: A Review ».

[10] Dogan, S., A. H. Sadka et A. M. Kondoz. 1999. « Efficient MPEG-4/H/263 video transcoder for interoperability of heterogeneous multimedia networks ». *Electronics Letters,* vol. 35, n° 11, p. 863-864.

[11] Feamster, Nick, et Susie Wee. 1999. « An MPEG-2 to H.263 transcoder ». In *Multimedia Systems and Applications II* (USA, 20-22 Sept. 1999). Vol. 3845, p. 164-75. SPIE-Int. Soc. Opt. Eng.

[12] Fernandez-Escribano, Gerardo, Jens Bialkowski, Jose A. Gamez, Hari Kalva, Pedro Cuenca, Luis Orozco-Barbosa et Andre Kaup. 2008. « Low-complexity heterogeneous video transcoding using data mining ». *IEEE Transactions on Multimedia,* vol. 10, n° 2, p. 286-299.

[13] Hur, Jae-Ho, Hyouk-Kyun Kwon et Yung-Lyul Lee. 2006. « H.264/AVC baseline profile to MPEG-4 visual simple profile transcoding to reduce the spatial resolution ». *International Journal of Imaging Systems and Technology,* vol. 16, n° 1, p. 24-33.

[14] Hur, Jae-Ho, et Yung-Lyul Lee. 2007. « H.264 to MPEG-4 resolution reduction transcoding ». In *TENCON 2005 - 2005 IEEE Region 10 Conference* (Melbourne, Australia, Nov 21-24 2005). Vol. 2007, p. 4084886. Institute of Electrical and Electronics Engineers Inc., Piscataway, NJ 08855-1331, United States.

[15] Hur, Jae-Ho, et Yung-Lyul Lee. 2007. « H.264 to MPEG-4 transcoding using block type information ». In *TENCON 2005 - 2005 IEEE Region 10 Conference* (Melbourne, Australia, Nov 21-24 2005). Vol. 2007, p. 4084887. Institute of Electrical and Electronics Engineers Inc., Piscataway, NJ 08855-1331, United States.

[16] Hwang, Jenq-Neng, et Tzong-Der Wu. 1998. « Motion vector re-estimation and dynamic frame-skipping for video transcoding ». In *Proceedings of the 1998 32nd Asilomar Conference on Signals, Systems & Computers. Part 2 (of 2)* (Pacific Grove, CA, Nov 1-4 1998). Vol. 2, p. 1606-1610. IEEE Comp Soc, Los Alamitos, CA, USA.

[17] « Integrated Performance Primitives ». 2010. Intel® Incorporated. <http://software.intel.com/sites/products/collateral/hpc/ipp/ipp_brief.pdf>. Consulté le 28 mai 2010.

[18] Intel. 2010. *Intel® Integrated Performance Primitives*, version. 5.3. Intel® Incorporated.

[19] International Organisation for Standardisation/International Electrotechnical Commission (ISO/IEC). 2000. *Information technology -- Generic coding of moving pictures and associated audio information: Video.* ISO/IEC 13 818-2:2000. Geneva, Switzerland,

<http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=31539>. Consulté le 18 décembre 2008.

[20] International Organisation for Standardisation/International Electrotechnical Commission (ISO/IEC). 2004. *Information technology -- Coding of audio-visual objects -- Part 2: Visual.* ISO/IEC 14496-2:2004. Geneva, Switzerland, <http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=39259>. Consulté le 18 décembre 2008.

[21] ITU Telecommunication Standardization Sector. 2005. *Video coding for low bit rate communication.* IUT-T Recommendation H.263 (01/2005). Geneva, Switzerland: International Telecommunication Union, 214 p. <http://www.itu.int/rec/T-REC-H.263-200501-I/en>. Consulté le 18 décembre 2008.

[22] ITU Telecommunication Standardization Sector. 2008. *Advanced video coding for generic audiovisual services.* IUT-T Recommendation H.264 (11/2007). Geneva, Switzerland: International Telecommunication Union, 540 p. <http://www.itu.int/rec/T-REC-H.264-200711-I/en>. Consulté le 17 décembre 2008.

[23] Jeongnam, Youn, Sun Ming-Ting et Lin Chia-Wen. 1999. « Motion vector refinement for high-performance transcoding ». *IEEE Transactions on Multimedia,* vol. 1, n° 1, p. 30-40.

[24] Kai-Tat, Fung, et Siu Wan-Chi. 2005. « Low complexity H.263 to H.264 video transcoding using motion vector decomposition ». In *IEEE International Symposium on Circuits and Systems (ISCAS)* (Piscataway, NJ, 23-26 May 2005). Vol. 2, p. 908-11. IEEE.

[25] Kalva, Hari. 2006. « Standards: The H.264 video coding standard ». *IEEE Multimedia,* vol. 13, n° 4, p. 86-90.

[26] Lee, Yung-Ki, Seong-Seon Lee et Yung-Lyul Lee. 2007. « MPEG-4 to H.264 transcoding with frame rate reduction ». *Multimedia Tools and Applications,* vol. 35, n° 2, p. 147-162.

[27] Lee, Yung-Ki, Sung-Sun Lee et Yung-Lyul Lee. 2006. « MPEG-4 to H.264 transcoding using macroblock statistics ». In *2006 IEEE International Conference on Multimedia and Expo, ICME 2006* (Toronto, ON, Canada, Jul 9-12 2006). Vol. 2006, p. 57-60. Institute of Electrical and Electronics Engineers Computer Society, Piscataway, NJ 08855-1331, United States.

[28] Lee, Yung-Ki, et Yung-Lyul Lee. 2007. « MPEG-4 to H.264 transcoding ». In *TENCON 2005 - 2005 IEEE Region 10 Conference* (Melbourne, Australia, Nov 21-24 2005). Vol. 2007, p. 4084891. Institute of Electrical and Electronics Engineers Inc., Piscataway, NJ 08855-1331, United States.

[29] Li, Ze-Nian, et Mark S. Drew. 2004. *Fundamentals of Multimedia*, 1st ed. United States: Prentice-Hall, 576 p.

[30] Liang, Yongfang, Fehmi Chebil et Asad Islam. 2006. « Compressed domain transcoding solutions for MPEG-4 visual simple profile and H.263 baseline videos in 3GPP services and applications ». *IEEE Transactions on Consumer Electronics,* vol. 52, n° 2, p. 507-514.

[31] Liang, Yongfang, Xiaohui Wei, Ishfaq Ahmad et Vishwanathan Swaminahan. 2007. « MPEG-4 to H.264/AVC transcoding ». In *IWCMC 2007: 2007 International Wireless Communications and Mobile Computing Conference* (Honolulu, HI, Aug 12-16 2007). p. 689-693. Association for Computing Machinery, New York, NY 10036-5701, United States.

[32] List, Peter, Anthony Joch, Jani Lainema, Gisle Bjontegaard et Marta Karczewicz. 2003. « Adaptive deblocking filter ». *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 13, n° 7, p. 614-619.

[33] Liu, Yu, Guiling Li, Qiang Tang et JiChang Guo. 2003. « DCT domain logo insertion of MPEG-2 transcoding ». In *CCECE 2003 Canadian Conference on Electrical and Computer Engineering: Toward a Caring and Humane Technology* (Montreal, Canada, May 4-7 2003). Vol. 2, p. 1219-1222. Institute of Electrical and Electronics Engineers Inc.

[34] Meng, Jianhao, et Shih-Fu Chang. 1998. « Embedding visible video watermarks in the compressed domain ». In *Proceedings of the 1998 International Conference on Image Processing, ICIP. Part 1 (of 3)* (Chicago, IL, Oct 4-7 1998). Vol. 1, p. 476-477. IEEE Comp Soc, Los Alamitos, CA, USA.

[35] Metoevi, Isabelle, et Stephane Coulombe. 2009. « Efficient MPEG-4 to H.264 transcoding exploiting MPEG-4 block modes, motion vectors, and residuals ». In., p. 224-229. Coll. « 2009 9th International Symposium on Communications and Information Technology, ISCIT 2009 ». Icheon, Korea, Republic of: IEEE Computer Society. <http://dx.doi.org/10.1109/ISCIT.2009.5341253>.

[36] Nakajima, Yasujuki, Hironao Hori et Tamotsu Kanoh. 1995. « Rate conversion of MPEG coded video by re-quantization process ». In *Proceedings of the 1995 IEEE International Conference on Image Processing. Part 3 (of 3)* (Washington, DC, Oct 23-26 1995). Vol. 3, p. 408-411. IEEE, Los Alamitos, CA, USA.

[37] Panusopone, Krit, Xuemin Chen et Fan Ling. 2001. « Logo insertion in MPEG transcoder ». In *2001 IEEE Interntional Conference on Acoustics, Speech, and Signal Processing* (Salt Lake, UT, May 7-11 2001). Vol. 2, p. 981-984. Institute of Electrical and Electronics Engineers Inc.

[38] Peng, Zhang, Huang Qing-Ming et Gao Wen. 2004. « Key techniques of bit rate reduction for H.264 streams ». In., p. 985-92. Coll. « Advances in Multimedia Information Processing - PCM 2004. 5th Pacific Rim Conference on Multimedia. Proceedings, Part II (Lecture Notes in Computer Science Vol.3332) ». Berlin, Germany: Springer-Verlag.

[39] Porter, Thomas, et Tom Duff. 1984. « Compositing digital images ». In *SIGGRAPH '84 Conference Proceedings* (Minneapolis, MN, 23-27 July 1984). Vol. 18, p. 253-9.

[40] Qiang, Li, Liu Xiaodong et Dai Qionghai. 2007. « Motion information exploitation in H.264 frame skipping transcoding ». In *9th International Conference on Advanced Concepts for Intelligent Vision Systems, ACIVS 2007* (Delft, Netherlands, Aug 28-31 2007). Vol. 4678 NCS, p. 768-776. Springer Verlag, Heidelberg, D-69121, Germany.

[41] Roma, Nuno, et Leonel Sousa. 2002. « Insertion of irregular-shaped logos in the compressed DCT domain ». In *Proceedings of 14th International Conference on Digital Signal Processing (DSP2002)* (Santorini, Greece, 1-3 July 2002). Vol. 1, p. 125-8. IEEE.

[42] Shanableh, Tamer, et Mohammed Ghanbari. 2000. « Heterogeneous video transcoding to lower spatio-temporal resolutions and different encoding formats ». *IEEE Transactions on Multimedia,* vol. 2, n° 2, p. 101-110.

[43] Shanableh, Tamer, et Mohammed Ghanbari. 2001. « Transcoding architectures for DCT-domain heterogeneous video transcoding ». In *IEEE International Conference on Image Processing (ICIP) 2001* (Thessaloniki, Oct 7-10 2001). Vol. 1, p. 433-436. Institute of Electrical and Electronics Engineers Computer Society.

[44] Shanableh, Tamer, et Mohammed Ghanbari. 2003. « Hybrid DCT/pixel domain architecture for heterogeneous video transcoding ». *Signal Processing: Image Communication,* vol. 18, n° 8 SPEC, p. 601-620.

[45] Shen, Huifeng, Xiaoyan Sun, Feng Wu, Houqiang Li et Shipeng Li. 2006. « A fast downsizing video transcoder for H.264/AVC with rate-distortion optimal mode decision ». In *2006 IEEE International Conference on Multimedia and Expo, ICME 2006* (Toronto, ON, Canada, Jul 9-12 2006). Vol. 2006, p. 2017-2020. Institute of Electrical and Electronics Engineers Computer Society, Piscataway, NJ 08855-1331, United States.

[46] Shin, Il-Hong, Yung-Lyul Lee et Hyunwook Park. 2004. « Motion estimation for frame-rate reduction in H.264 transcoding ». In *Proceedings - Second IEEE Workshop on Software Technologies for Future Embedded and Ubiquitous Systems,*

May 11-12 2004 (Vienna, Austria). p. 63-67. IEEE Computer Society, Los Alamitos, CA 90720-1314, United States.

[47] Sun, Huifang, Wilson Kwok et Joel W. Zdepski. 1996. « Architectures for MPEG compressed bitstream scaling ». *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 6, n° 2, p. 191-199.

[48] Tang, Qiang, Hassan Mansour, Panos Nasiopoulos et Rabab Ward. 2008. « Bit-rate estimation for bit-rate reduction H.264/AVC video transcoding in wireless networks ». In., p. 464-467. Coll. « 3rd International Symposium on Wireless Pervasive Computing, ISWPC 2008, Proceedings ». Santorini, Greece: Inst. of Elec. and Elec. Eng. Computer Society. <http://dx.doi.org/10.1109/ISWPC.2008.4556251>.

[49] Vetro, Anthony, Charilaos Christopoulos et Huifang Sun. 2003. « Video transcoding architectures and techniques: An overview ». *IEEE Signal Processing Magazine,* vol. 20, n° 2, p. 18-29.

[50] Vetro, Anthony, Peng Yin, Bede Liu et Huifang Sun. 2002. « Reduced spatio-temporal transcoding using an intra refresh technique ». In *2002 IEEE International Symposium on Circuits and Systems* (Phoenix, AZ, May 26-29 2002). Vol. 4, p. 723-726. Institute of Electrical and Electronics Engineers Inc.

[51] Wee, Susie J., John G. Apostolopoulos et Nick Feamster. 1999. « Field-to-frame transcoding with spatial and temporal downsampling ». *IEEE International Conference on Image Processing,* vol. 4, p. 271-275.

[52] Wiegand, Thomas, Heiko Schwarz, Anthony Joch, Faouzi Kossentini et Gary J. Sullivan. 2003. « Rate-constrained coder control and comparison of video coding standards ». *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 13, n° 7, p. 688-703.

[53] Wiegand, Thomas, Gary J. Sullivan, Gisle Bjontegaard et Ajay Luthra. 2003. « Overview of the H.264/AVC video coding standard ». *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 13, n° 7, p. 560-576.

[54] Xin, Jun, Chia-Wen Lin et Ming-Ting Sun. 2005. « Digital video transcoding ». *Proceedings of the IEEE,* vol. 93, n° 1, p. 84-97.

[55] Xin, Jun, Ming-Ting Sun et Kangwook Chun. 2002. « Motion Re-estimation for MPEG-2 to MPEG-4 Simple Profile Transcoding ». In *Int. Packet Video Workshop Pittsburgh,* (PA, 2002).

[56] Yin, Peng, Min Wu et Bede Liu. 2000. « Video transcoding by reducing spatial resolution ». In *International Conference on Image Processing (ICIP 2000)*

(Vancouver, BC, Sep 10-13 2000). Vol. 1, p. 972-975. Institute of Electrical and Electronics Engineers Computer Society.

[57] Youn, Jeongnam, Ming-Ting Sun et Jun Xin. 1999. « Video transcoder architectures for bit rate scaling of H.263 bit streams ». In *Proceedings of the 1999 7th International Multimedia Conference - ACM MULTIMEDIA '99, Oct 30-Nov 5 1999* (New York, NY). p. 243-250. ACM.

[58] Youn, Jeongnam, Jun Xin et Ming-Ting Sun. 2000. « Fast video transcoding architectures for networked multimedia applications ». In *Proceedings of the IEEE 2000 Internaitonal Symposium on Circuits and Systems* (Geneva, Switz, May 28-May 31 2000). Vol. 4, p. 25-28. Institute of Electrical and Electronics Engineers Inc., Piscataway, NJ, USA.

[59] Zhang, Peng, Yan Lu, Qingming Huang et Wen Gao. 2004. « Mode mapping method for H.264/AVC spatial downscaling transcoding ». In *2004 International Conference on Image Processing, ICIP 2004* (Singapore, Oct 18-21 2004). Vol. 4, p. 2781-2784. Institute of Electrical and Electronics Engineers Computer Society, Piscataway, NJ 08855-1331, United States.

[60] Zhu, Wenwu, Kyeong Ho Yang et Marc J. Beacken. 1998. « CIF-to-QCIF video bitstream down-conversion in the DCT domain ». *Bell Labs Technical Journal,* vol. 3, n° 3, p. 21-29.